BASIC PROGRAMS FOR THE ATARI® 600XL & 800XL

BY TIMOTHY ORR KNIGHT,
PROGRAMS BY DARREN LaBATT

No. 1726
$14.95

BASIC PROGRAMS
FOR THE
ATARI®
600XL & 800XL

BY TIMOTHY ORR KNIGHT,
PROGRAMS BY DARREN LaBATT

TAB BOOKS Inc.
BLUE RIDGE SUMMIT, PA 17214

To my grandparents, William T. and Willie Lee Rucker and Delos L. and Ruth V. Knight, whose heritage, strength, and wisdom have enriched my life.

FIRST EDITION

FIRST PRINTING

Library of Congress Cataloging in Publication Data

Knight, Timothy Orr.
BASIC programs for the ATARI 600XL and 800XL.

Includes index.
1. Atari 600XL (Computer)—Programming. 2. Atari 800XL (Computer)—Programming. 3. Basic (Computer program language) I. LaBatt, Darren. II. Title. III. Title: B.A.S.I.C. programs for the Atari six hundred XL and eight hundred XL.
QA76.8.A82K65 1984 001.64'25 83-24243
ISBN 0-8306-0726-9
ISBN 0-8306-1726-4 (pbk.)

Cover photograph courtesy of Atari, Inc.

Contents

Other TAB Books by the Authors

No. 1706 *Using and Programming the ADAM™, including Ready-to-Run Programs* (by Timothy Orr Knight)
No. 1716 *Basic BASIC Programs for the ADAM™*

Introduction

This book is designed to help you learn more about your ATARI 600XL or 800LX home computer, provide you with a number of programs that will be useful to you now or in the near future, and demonstrate to you how a program evolves from start to finish. The "method of teaching" in this book, however, is not a description of the BASIC language, nor is it a summary of how BASIC works and how you may use it. This book is a collection of programs for your ATARI home computer. By using these programs, you will learn the things mentioned above.

When you are reading this book, you "participate" by typing in the program lines as they are shown to you. Before you type a new program into your ATARI, type NEW followed by (as always) a press of the <RETURN> key. This will prepare the computer for your program. Then you can begin reading about the program, how it was made, and what the lines mean as you enter them into the machine. When you have finished typing in the program, you will have a ready-to-run piece of software. You will also know how that program works.

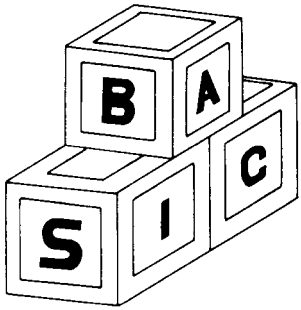
By the time you reach the end of this book, I hope you will have accomplished three things:

■ You will have finished typing in all 20 programs contained in this book, so that you can use them later and perhaps modify them for your own purposes.

■ You will understand how each of the programs works and how it was developed into a working piece of software.
■ You will comprehend the method by which a program is made, from idea to completion.

As you read this book, keep in mind that this should be a learning experience as well as an entertaining one. You'll soon discover that these two elements combine to increase your experience and your knowledge of the ATARI home computers.

Chapter 1



Programming with Your ATARI Computer

This is a book designed to help you in a number of ways. First of all, it is a book full of useful, educational, and entertaining programs which will be in ready-to-run form once you have purased this book. In fact, with this book you are also getting a large number of programs which would normally cost you quite a bit of money at a software store. Second, this is a book which will help teach you how to use your ATARI home computer effectively, since you are the person who will be typing in the programs, learning about the programs as you type them into the computer, and running these programs to see the result of your efforts. And, last, this book is intended to teach you a method of programming and explain how you can make your own programs—and perhaps even sell them for a profit.

To begin, let's take a look at the setup or format of the programs and instructions within it. This format is easy to understand and was designed to help programmers at any level, beginning or more advanced, to understand the contents of this book.

THE FORMAT OF THIS BOOK

This book centers around the 20 programs it contains. Each is thoroughly explained; you will find three types of explanations for each program. First of all, I will explain the purpose of program lines as I introduce them to you. Then you should type

them into the computer. Don't use the program until you have completely finished typing it into the computer, and understand how it works. Another explanation provided is the explanation of the important lines within the program. I will tell you the main sets of program lines and will briefly describe the purpose of that group of lines. Finally, I will describe the purpose of each of the variables in the program.

To illustrate this more clearly, let's assume that I have made a very simple program for the book, as shown below:

```
10  REM THIS PROGRAM SAYS HELLO TO A PERSON
20  PRINT"HI THERE. WHAT IS YOUR NAME";
30  INPUT NAME$
40  PRINT"HELLO, ";NAME$
50  GOTO 20
```

This is an extremely simple program, but it will serve our purposes for this illustration. Now that I have created this program, I would describe it to you in the book in a format that might look like this:

This is a simple program which you may use to let people become familiar and feel more at ease with the ATARI home computer. To begin the program, type in the REMark statement to state the purpose of the program:

```
10  REM THIS PROGRAM SAYS HELLO TO A PERSON
```

Next, the computer will print its greeting and accept input from a person to find out their name.

```
20  PRINT"HI THERE. WHAT IS YOUR NAME";
30  INPUT NAME$
```

Then the computer will greet that person by name and repeat the whole "welcoming process" by going back to line 20:

```
40  PRINT"HELLO, ";NAME$
50  GOTO 20
```

You might want to expand on this program by programming the computer to get more information about the person, or

make comments such as "YOU HAVE A NICE NAME" or "YOUR BIRTHDAY IS LATE IN THE YEAR, ISN'T IT?" based on the input from the user.

Important Line Numbers in HELLO

10	*Tells the purpose of the program*
20-30	*Greets the person and gets name*
40	*Greets the person by name*
50	*Goes back to line 20*

Important Variables in HELLO

NAME$	*Receives the name of the person*

From the above example, you can get a good idea as to how the book is set up. You will learn how the program was made, what purpose each of the lines serves, and what purpose the variables serve. As you might guess, this not only provides you with a good program, which you can easily understand and modify for your own purposes, but it is also an excellent and enjoyable way to learn programming.

THE PROGRAMS

To give you an idea of what you have to look forward to in this book, here are the 19 other programs, with a brief description of each.

Sounds and Music

Song Library. This is a collection of four songs for your ATARI to play, including "The Caisson Song," "Frère Jacques," "My Bonnie," and Chopin's *fantasie impromptu*.

Sound Maker. Using your ATARI's four voices, you can create your own sounds with this program.

Sound Effects Library. This program provides a collection of effects for your ATARI to play, including lasers, alarms, gunshots, and spaceship blast-off.

Music Creator. Writing music on your ATARI is easier with this program, which allows you to input notes and their lengths.

Education

French Tutor. This is a tutorial program which teaches you

and tests you on a vocabulary of simple French words.

States and Capitals. Learn states and capitals by matching them in this program, which also finds information for you.

History Quiz. Who was Herodotus? Find out with this program, which tests you on various events and people in history.

Spelling Tester. Pick the incorrect spelling of a word from the possibilities presented by this useful program.

Mathematics

Geometry. Find the answer to 16 kinds of geometry problems, such as areas of figures and volumes of solids, with this handy program.

Calculator. Find answers to basic math, trigonometric, and square root problems using this program, the "computer calculator."

Basic Skills Checkup. Test yourself on addition, subtraction, and multiplication.

Length Converter. Use this program to convert from metric to "standard," or from any metric or standard unit to any other metric or standard unit.

Graphics

Drawer. Use the computer screen as a drawing easel for your visual creations.

Graph. The computer makes a line graph of data you supply to the program.

Shapes. This program shows you some of the shapes the ATARI can make and describes how they were made.

Colorbar. Learn the effects of luminance (brightness) and hue (color) as you create a wide assortment of colors on the screen.

Games

Blackjack. Play this classic against the computer.

Guess My Number. This is a simple game which teaches you a lot about how the computer's random number generator operates.

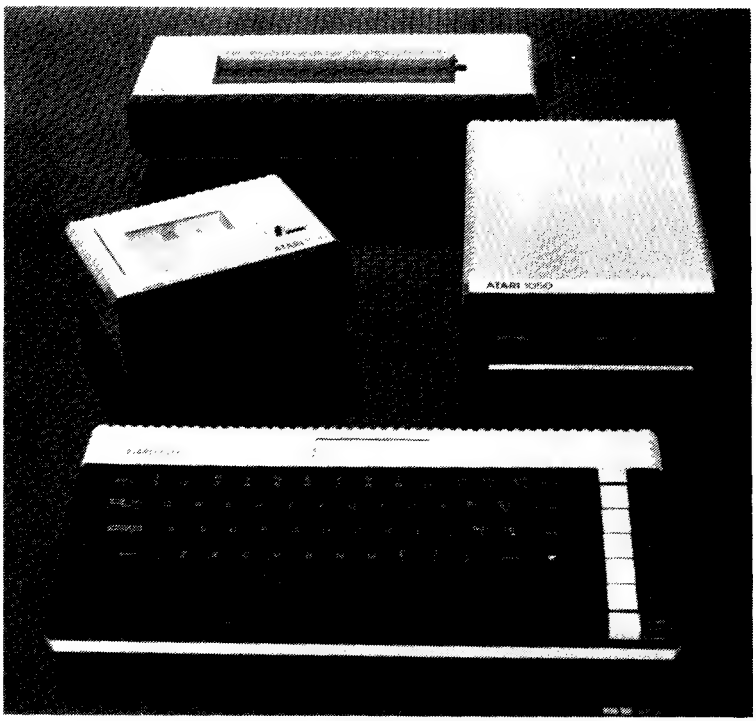
Simon Says. Memorize as many colors and sounds as you can to repeat to the computer.

These programs will get you off to a good start on making your own programs, and will also provide you with a nice selection of software to use and enjoy later.

USING THIS BOOK

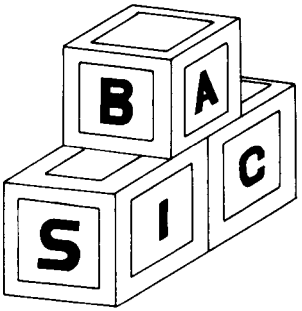
Using this book by simply typing in the programs and running them is simple enough (and perfectly all right), since that is the main point. An extension of this concept, however, is to learn how to create program ideas and programs yourself. By examining these programs and discovering how they were created, you can develop your own system for creating an idea for a program, developing it into a program, and fixing any problems ("bugs") in it, so that it can be a useful addition to a computer system.

While you are studying and typing in these programs, write down any ideas you might have for your own programs and begin working on them whenever you feel you are ready. Your experience with your programs will be a far better teacher than this book, since an individual is his or her own best teacher. Nevertheless, I hope this book will get you off to a good start, so you can begin using your own creativity and logic to make your computer an extension of your mind.



The ATARI 600XL computer system (courtesy Atari, Inc.)

Chapter 2



Sound and Music Programs

The ATARI 600XL and 800XL home computers (Figs. 2-1 and 2-2) come equipped with four separate voices with which you can create music and sounds. To do this, you specify in your programs which voice (sound channel) you want to use, what sound frequency that channel should produce, and what the volume of that sound should be. In this chapter we'll examine and create four different programs which produce sounds and music, and which let you program different songs and sounds into your ATARI.

SONG LIBRARY

To help stir your interest in the ATARI's power to create sounds and music, here is a program which will play any one of four different available songs for you. Simply select the song you want to hear and let it play.

In the program, the ATARI first clears the screen then displays the different songs which can be played. You are then allowed to give the computer your choice.

```
100  GRAPHICS 0

105  RESTORE

110  POSITION 9,6

120  PRINT"1. CAISSON SONG"
```

```
130  POSITION 9,7
140  PRINT"2. FRERE JACQUES"
150  POSITION 9,8
160  PRINT"3. FANTASIE IMPROMPTU"
170  POSITION 9,9
180  PRINT"4. MY BONNIE"
190  POSITION 9,10
200  PRINT"5. END"
210  PRINT:PRINT:PRINT:INPUT CHOICE
```

Next, the screen is cleared again and the ATARI goes to the song you wish to be played or, if you selected the fifth option, the computer ends the program by going to line 6000.

```
220  GRAPHICS 0
230  ON CHOICE GOTO 1000,2000,3000,4000,6000
```

At this point in the program, the computer will access one of the four routines below to play a particular song. Note that the number of frequency/length combinations (note data) varies, since the lengths of the songs vary.

```
1000 FOR LOOP1=1 TO 83
1020 READ A,B
1030 SOUND 1,A,14,15
1040 FOR LN=1 TO B*1.5:NEXT LN
1050 SOUND 1,0,0,0
1060 NEXT LOOP1
1070 GOTO 100
2000 FOR LOOP2=1 TO 83
2010 READ A,B
2020 NEXT LOOP2
2030 FOR LOOP1=1 TO 64
2040 READ A,B
```

```
2050 SOUND 1,A,14,15
2060 FOR LN=1 TO B*1.7:NEXT LN
2070 SOUND 1,0,0,0
2080 NEXT LOOP1
2090 GOTO 100
3000 FOR LOOP2=1 TO 147
3010 READ A,B
3020 NEXT LOOP2
3030 FOR LOOP1=1 TO 61
3040 READ A,B
3050 SOUND 1,A,14,15
3060 FOR LN=1 TO B*1.3:NEXT LN
3070 SOUND 1,0,0,0
3080 NEXT LOOP1
3090 GOTO 100
4000 TIMES=0
4010 FOR LOOP2=1 TO 208
4020 READ A,B
4030 NEXT LOOP2
4040 FOR LOOP1=1 TO 60
4045 IF LOOP1=35 THEN GOSUB 4200
4050 READ A,B
4060 SOUND 1,A,14,15
4070 FOR LN=1 TO B*1.2:NEXT LN
4080 SOUND 1,0,0,0
4090 NEXT LOOP1
4100 TIME=TIMES+1
4110 IF TIMES=1 THEN RESTORE:GOTO 4010
4120 GOTO 100
4200 FOR REST=1 TO 70:NEXT REST
4210 RETURN
```

The data below contains the frequencies and the lengths of the notes in all of the songs:

```
5000 DATA 81,50,96,50,81,100,81,50,96,50,81,100
5010 DATA 81,50,96,50,81,75,72,25,81,50,96,50
5020 DATA 81,100,96,50,91,50,81,50,91,100,108,50
5030 DATA 81,50,91,100,108,50,125,300,81,50,96,50
5040 DATA 81,100,81,50,96,50,81,100,81,50,96,50
5050 DATA 81,75,72,25,81,50,96,50,81,100,96,50
5060 DATA 91,50,81,50,91,100,108,50,81,50,91,100
5070 DATA 108,50,125,300,81,50,81,50,63,100,63,100
5080 DATA 81,100,81,50,81,50,72,50,64,50
5090 DATA 63,50,72,50,81,100,63,50,63,100,64,50
5100 DATA 72,50,64,50,63,50,72,50,53,300,81,50
5110 DATA 81,50,63,100,63,100,64,100,72,50,64,50
5120 DATA 63,50,72,50,81,100,96,50,91,50,81,50
5130 DATA 91,100,108,50,81,50,91,100,108,50,125,300
5200 DATA 125,50,108,50,96,50,125,50,126,50,108,50
5210 DATA 96,50,125,50,96,50,91,50,81,100,96,50
5220 DATA 91,50,81,100,81,27,72,27,81,27,91,27
5230 DATA 96,50,125,50,81,27,72,27,81,27,91,27
5240 DATA 96,50,125,50,125,50,162,50,126,100,126,50
5250 DATA 162,50,125,100,91,50,81,50,72,50,91,50
5260 DATA 91,50,81,50,72,50,91,50,72,50,68,50
5270 DATA 62,100,72,50,68,50,62,100,62,27,53,27
5280 DATA 62,27,68,27,72,50,91,50,62,27,53,27
5290 DATA 62,27,68,27,72,50,91,50,91,50,125,50
5300 DATA 91,100,91,50,125,50,91,100
5400 DATA 81,200,72,50,81,50,63,50,53,50,47,200
5410 DATA 40,200,45,100,47,100,53,100,47,50,63,50
5420 DATA 81,400,72,200,68,50,72,50,53,50,47,50
```

```
5430 DATA 45,100,47,100,53,100,47,100,63,100,64,27
5440 DATA 63,27,53,27,63,27,47,150,53,50,53,400
5450 DATA 81,200,72,50,81,50,63,50,53,50,47,200
5460 DATA 40,200,45,100,47,100,53,100,47,50,63,50
5470 DATA 81,400,72,200,68,50,72,50,53,50,47,50
5480 DATA 45,100,47,100,53,100,47,100,63,100,64,27
5490 DATA 63,27,53,27,63,27,47,150,53,50,53,200,63,200
5500 DATA 108,50,64,75,72,25,81,50,72,50,81,20
5510 DATA 96,50,108,50,128,200,108,50,64,75,72,25
5520 DATA 81,50,81,50,85,50,81,50,72,250,108,50
5530 DATA 64,75,72,25,81,50,72,50,81,50,96,50
5540 DATA 108,50,128,200,108,50,96,50,72,50,81,50
5550 DATA 85,50,96,50,85,50,81,250,108,150,81,150
5560 DATA 96,150,72,100,81,50,85,50,85,50,85,50
5570 DATA 85,50,96,50,85,50,81,100,72,50,64,150
5580 DATA 108,150,81,150,96,150,72,100,81,50,85,50
5590 DATA 85,50,85,50,85,50,96,50,85,50,81,200
```

Finally, line 6000 clears the screen and lets the program end.

```
6000 GRAPHICS 0
```

In order to add your own songs to this program, add a new option to the menu, compensate for your addition in line 230 (the ON . . . GOTO statement), put in new program lines to read and play the song, and add the DATA statements for the frequencies and the lengths of the notes in your song.

Important Variables in Song Library

CHOICE	Input for choice of song
LOOP1	Reads notes for songs
LOOP2	Reads unneeded notes
LN	Delay loop for note length

A	Note value (period)
B	Note length
TIMES	Used to play "My Bonnie" twice
REST	Rest in music

Important Lines in Song Library

100-200	Sets up menu
210-230	Gets choice of song
1000-1070	Plays "Caisson Song"
2000-2090	Plays "Frère Jacques"
3000-3090	Plays *fantasie impromptu*
4000-4210	Plays "My Bonnie"
5000-5590	Data
6000	Ends program

SOUND MAKER

You have the ability to create unique notes and sounds on your ATARI; this program will make that task much easier for you. This program allows you to change the volume and the frequency of any of the four voices on the ATARI, and you will hear all of them playing simultaneously. In addition, you will see a graphic representation of the frequencies on your screen, along with the volumes and the frequencies printed at the bottom of the screen.

In order to use this program, use the following commands:

q,w,e,r	Increase the frequency of voice 1,2,3, or 4 (respectively) by one relative unit in a range from 0 to 255.
Q,W,E,R	Decrease the frequency of voice 1,2,3, or 4 by one unit.
a,s,d,f	Increase the frequency of voice 1,2,3, or 4 by ten units.
A,S,D,F	Decrease the frequency of voice 1,2,3, or 4 by ten units.
z,x,c,v	Increase the volume by one relative unit.
Z,X,C,V	Decrease the volume by one relative unit.

The first part of the program is used to establish the vari-

ables of Sound Maker and to clear the value at location 764 in memory, the byte at which information from the keyboard is registered.

```
100  GRAPHICS 14
110  P1=255:P2=255:P3=255:P4=255
120  V1=0:V2=0:V3=0:V4=0
125  POKE 764,255
```

The computer begins monitoring the keyboard. Below are the IF-THEN statements for increasing or decreasing the frequency of a voice by one.

```
130  KEY=PEEK(764)
140  IF KEY=47 THEN P1=P1+1:GOTO 1000
150  IF KEY=46 THEN P2=P2+1:GOTO 1000
160  IF KEY=12 THEN P3=P3+1:GOTO 1000
170  IF KEY=40 THEN P4=P4+1:GOTO 1000
180  IF KEY=111 THEN P1=P1-1:GOTO 1000
190  IF KEY=110 THEN P2=P2-1:GOTO 1000
200  IF KEY=106 THEN P3=P3-1:GOTO 1000
210  IF KEY=104 THEN P4=P4-1:GOTO 1000
```

Here are the lines which increase or decrease the frequency of a voice by ten.

```
220  IF KEY=63 THEN P1=P1+10:GOTO 1000
230  IF KEY=62 THEN P2=P2+10:GOTO 1000
240  IF KEY=58 THEN P3=P3+10:GOTO 1000
250  IF KEY=56 THEN P4=P4+10:GOTO 1000
260  IF KEY=127 THEN P1=P1-10:GOTO 1000
270  IF KEY=126 THEN P2=P2-10:GOTO 1000
280  IF KEY=122 THEN P3=P3-10:GOTO 1000
290  IF KEY=120 THEN P4=P4-10:GOTO 1000
```

Finally, here are the lines (in addition to line 400) which are

used to increase or decrease the volume of a voice. These make the computer go back to line 130 to repeat the "keyboard scanning" process.

```
300  IF KEY=23 THEN V1=V1+1:GOTO 1000
310  IF KEY=22 THEN V2=V2+1:GOTO 1000
320  IF KEY=18 THEN V3=V3+1:GOTO 1000
330  IF KEY=16 THEN V4=V4+1:GOTO 1000
340  IF KEY=87 THEN V1=V1-1:GOTO 1000
350  IF KEY=86 THEN V2=V2-1:GOTO 1000
360  IF KEY=82 THEN V3=V3-1:GOTO 1000
370  IF KEY=80 THEN V4=V4-1:GOTO 1000
400  GOTO 130
```

I then programmed the computer to check the values for the frequencies and volumes of all of the voices to see if any were "out of range." If so, the computer would make the proper correction.

```
1000 IF P1>255 THEN P1=255:GOTO 130
1010 IF P2>255 THEN P2=255:GOTO 130
1020 IF P3>255 THEN P3=255:GOTO 130
1030 IF P4>255 THEN P4=255:GOTO 130
1040 IF P1<2 THEN P1=2:GOTO 130
1050 IF P2<2 THEN P2=2:GOTO 130
1060 IF P3<2 THEN P3=2:GOTO 130
1070 IF P4<2 THEN P4=2:GOTO 130
1080 IF V1>15 THEN V1=15:GOTO 130
1090 IF V2>15 THEN V2=15:GOTO 130
1100 IF V3>15 THEN V3=15:GOTO 130
1110 IF V4>15 THEN V4=15:GOTO 130
1120 IF V1<0 THEN V1=0:GOTO 130
1130 IF V2<0 THEN V2=0:GOTO 130
1140 IF V3<0 THEN V3=0:GOTO 130
1150 IF V4<0 THEN V4=0:GOTO 130
```

The ATARI then plays the sounds through all four of its voices:

```
2000 SOUND 0,255-P1,14,V1
2100 SOUND 1,255-P2,14,V2
2200 SOUND 2,255-P3,14,V3
2300 SOUND 3,255-P4,14,V4
```

Immediately after playing the sounds, the ATARI displays the graphics representing the frequencies of the sounds.

```
2500 COLOR 0
2510 PLOT 25,0
2520 DRAWTO 25,(255-P1)/2
3000 COLOR 1
3010 PLOT 25,159
3020 DRAWTO 25,(255-P1)/2
3100 COLOR 0
3110 PLOT 30,0
3120 DRAWTO 30,160-(V1*10)
3200 COLOR 3
3210 PLOT 30,159
3220 DRAWTO 30,160-(V1*10)
3300 COLOR 0
3310 PLOT 50,0
3320 DRAWTO 50,(255-P2)/2
3400 COLOR 1
3410 PLOT 50,159
3420 DRAWTO 50,(255-P2)/2
3500 COLOR 0
3510 PLOT 55,0
3520 DRAWTO 55,160-(V2*10)
```

```
3600 COLOR 3
3610 PLOT 55,159
3620 DRAWTO 55,160-(V2*10)
3700 COLOR 0
3710 PLOT 85,0
3720 DRAWTO 85,(255-P3)/2
3800 COLOR 1
3810 PLOT 85,159
3820 DRAWTO 85,(255-P3)/2
3900 COLOR 0
3910 PLOT 90,0
3920 DRAWTO 90,160-(V3*10)
4000 COLOR 3
4010 PLOT 90,159
4020 DRAWTO 90,160-(V3*10)
4100 COLOR 0
4110 PLOT 115,0
4120 DRAWTO 115,(255-P4)/2
4200 COLOR 1
4210 PLOT 115,159
4220 DRAWTO 115,(255-P4)/2
4300 COLOR 0
4310 PLOT 120,0
4320 DRAWTO 120,160-(V4*10)
4400 COLOR 3
4410 PLOT 120,159
4420 DRAWTO 120,160-(V4*10)
```

Last, the computer shows the values of the frequencies and the volumes:

```
4500 PRINT
5000 PRINT 255-P1,255-P2,255-P3,255-P4
```

```
5010 PRINT V1,V2,V3,V4

6000 GOTO 125
```

Working with this program will produce some interesting effects. See if you can create certain tones and special sound effects for your programs by using this utility.

Important Variables in Sound Maker

P1,P2,P3,P4	Period values for voices one to four
V1,V2,V3,V4	Volumes for voices one to four
KEY	Keyboard input

Important Lines in Sound Maker

100-130	Initialization
140-400	Sees which key is pressed and adds or subtracts from voice or period
1000-1150	Makes sure new values are not out of range
2000-2300	Plays four sounds
2500-4420	Makes graphs for period and volume of all four voices
5000-5010	Prints period value and volume value
6000	Goes back to get more keyboard input

SOUND EFFECTS LIBRARY

For some "prepackaged" sounds, here is a selection of several sound effects. Simply select the one you would like to hear, and press S when you want to stop listening to the sound.

First, the screen is cleared, the menu is displayed, and your ATARI lets you input your choice.

```
50   GRAPHICS 0

100  POSITION 9,6

110  PRINT"SOUND EFFECTS"

120  POSITION 6,9

130  PRINT"1. LASER FIRE"

140  POSITION 6,10

150  PRINT"2. ALARM"
```

```
160  POSITION 6,11
170  PRINT"3. SPACESHIP THRUST"
180  POSITION 6,12
190  PRINT"4. REBOUND NOISE"
200  POSITION 6,13
210  PRINT"5. GUN SHOT"
220  POSITION 6,14
230  PRINT"6. RANDOM NOISE"
240  POSITION 6,15
250  PRINT"7. END"
260  PRINT:PRINT:PRINT"HIT 'S' TO END SOUND"
280  PRINT:PRINT:INPUT CHOICE
```

Next, the computer goes to the proper line number for your choice and plays the sound you want to hear.

```
290  ON CHOICE1 GOTO 300,400,500,600,700,800,900
300  FOR LOOP1=1 TO 50
310  SOUND 1,LOOP1+5,14,15
320  NEXT LOOP1
330  SOUND 1,0,0,0
340  FOR WAIT=1 TO 150:NEXT WAIT
350  GOSUB 3000
360  GOTO 300
400  PERIOD=255
410  FOR LOOP1=1 TO 50
420  SOUND 1,PERIOD,14,15
425  PERIOD=PERIOD-5
430  NEXT LOOP1
440  SOUND 1,0,0,0
450  FOR WAIT=1 TO 150:NEXT WAIT
460  GOSUB 3000
```

```
470  GOTO 400
500  FOR LOOP1=1 TO 100
510  SOUND 1,120,8,15
520  NEXT LOOP1
530  SOUND 1,0,0,0
540  GOSUB 3000
550  GOTO 500
600  VOL=15
610  FOR LOOP1=1 TO 15
620  SOUND 1,155,14,VOL
630  VOL=VOL-1
640  NEXT LOOP1
650  SOUND 1,0,0,0
660  FOR WAIT=1 TO 150:NEXT WAIT
670  GOSUB 3000
680  GOTO 600
700  VOL=15
710  FOR LOOP1=1 TO 3
720  SOUND 1,150,8,VOL
730  NEXT LOOP1
735  VOL=7
750  SOUND 1,255,6,VOL-LOOP2
760  NEXT LOOP2
770  SOUND 1,0,0,0
775  FOR WAIT=1 TO 150:NEXT WAIT
780  GOSUB 3000
790  GOTO 700
800  RN1=INT(255*RND(1)+1)
810  RN2=INT((28*RND(1)+1)/2)
820  RN3=INT(15*RND(1)+1)
```

```
830  SOUND 1,RN1,RN2,RN3

840  GOSUB 3000

850  GOTO 800

900  GRAPHICS 0:END
```

The last part of the program monitors the keyboard for the "S" key. If it is pressed, the computer stops playing the sound and returns to the menu. If not, the ATARI keeps on playing.

```
3000 IF PEEK(764)=62 THEN POKE 764,255:SOUND 1,0,0,0:

GOTO 100

3010 RETURN
```

Important Variables in Sound Effects Library

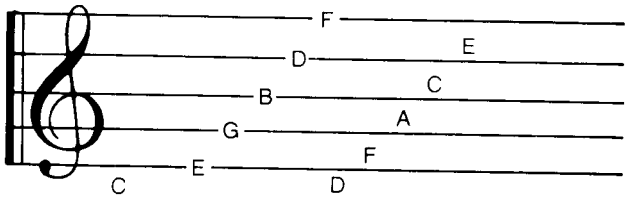
CHOICE	User's choice of sound
LOOP1	Used to make sounds
WAIT	Delay loop
PERIOD	The frequency value
VOL	Volume
LOOP2	Makes sound for second part of gunfire
RN1	Random value for period
RN2	Random value for distortion
RN3	Random value for volume

Important Lines in Sound Effects Library

50-260	Set up menu
280-290	Get choice, go to sound
300-360	Creates laser fire sound
400-470	Creates alarm sound
500-550	Spaceship thrust
600-680	Rebound noise
700-790	Gunshot noise
800-850	Random noise
900	Ends program
3000-3010	Turns off sound and resets the keyboard

MUSIC CREATOR

Not only may you want to create certain sounds on your computer, you might also want to write music with your ATARI.



SYMBOL	NOTE	VALUE
𝅝	Whole	4 beats
𝅗𝅥	Half	2 beats
♩	Quarter	1 beat
♪	Eighth	½ beat
𝅘𝅥𝅯	Sixteenth	¼ beat

Fig. 2-1. Musical notes.

The following program will allow you to enter notes and their lengths, and then chain the notes together to make a song.

Music creator first establishes the variables it is going to use, shows its title, and lets you begin inputting your song. To input a note, type in a four-character code consisting of:

1) A note name from the set A,B,C,D,E,F,G.
2) A blank character if the note is natural, or a sharp sign (♯) if it is sharp.
3) The octave level, a number from 1 to 3 representing how "high" the octave is.
4) The length of the note, which can be S (sixteenth note), E (eighth), Q (quarter), H (half), or W (whole note).

The notes available, as well as a comparison of their time values, are shown in Fig. 2-1.

```
50    DIM INP$(5):DIM A$(3)

60    DIM P(90):DIM L(90)
```

```
70   DIM L$(3)

100  GRAPHICS 0

110  POSITION 10,4

120  PRINT"MUSIC CREATOR"

130  NM=1

135  PRINT:PRINT

140  INPUT INP$
```

This section of the program lets your ATARI accept two other commands—PLAY, which will play your song, and END, which will stop the program completely.

```
150  IF INP$="PLAY" THEN NM=NM-1:GOTO 1000

160  IF INP$="END" THEN GRAPHICS 0:END
```

At this point, the computer will compare your note with the notes in the DATA statements at the end of the program to find out the frequency of the note you selected.

```
170  READ A$,B,C

200  IF A$=INP$(1,2) THEN 300

210  GOTO 150

300  D=VAL(INP$(3,3))

310  IF B<>D THEN GOTO 150

400  P(NM)=C

500  L$=INP$(4,4)
```

Then your ATARI determines the length of the note you selected:

```
510  IF L$="S" THEN L=25

520  IF L$="E" THEN L=50

530  IF L$="Q" THEN L=100

540  IF L$="H" THEN L=200

550  IF L$="W" THEN L=400

560  L(NM)=L
```

To let you know what you have just entered, the computer will play the note. It will then go back to the INPUT routine so you can continue supplying notes until you are ready to PLAY the song.

```
600  SOUND 1,P(NM),14,15
610  FOR LOOP1=1 TO L(NM):NEXT LOOP1
620  SOUND 1,0,0,0
700  NM=NM+1
710  RESTORE
720  GOTO 140
```

When you are ready to play the song, this routine will do just that.

```
1000 FOR LOOP2=1 TO NM
1010 SOUND 1,P(LOOP2),14,15
1020 FOR LOOP3=1 TO L(LOOP2):NEXT LOOP3
1030 NEXT LOOP2
2000 SOUND 1,0,0,0
2010 NM=NM+1
2020 GOTO 140
```

Here, at last, is the data for the notes you might use in the program:

```
5000 DATA C ,1,251,C#,1,230,D ,1,217
5010 DATA D#,1,204,E ,1,193,F ,1,182
5020 DATA F#,1,173,G ,1,162,G#,1,153
5030 DATA A ,1,144,A#,1,136,B ,1,128
5040 DATA C ,2,126,C#,2,114,D, 2,108
5050 DATA D#,2,102,E ,2,96,F ,2,91
5060 DATA F#,2,85,G ,2,81,G#,2,76
5070 DATA A ,2,72,A#,2,68,B ,2,64
5080 DATA C ,3,63,C#,3,57,D ,3,53
```

```
5090 DATA D#,3,50,E ,3,47,F ,3,45
5100 DATA F#,3,42,G ,3,40,G#,3,37
5110 DATA A ,3,35,A#,3,33,B ,3,31
```

See if you can construct a routine to save your songs to the disk. If you would like to add more songs to the "Song Library" program earlier in this chapter, add the songs you create with this program.

Important Variables in Music Creator

INP$	Input a note of music
A$	The note frequency
P(X)	The period for each note
L(X)	The length of each note
L$	The input for the note length
NM	Number of the note
B	Data for octave number
C	Data for frequency value
D	Numeric value of octave number
LOOP1	Delay loop for making note length
LOOP2	Used for playing song
LOOP3	Another delay loop

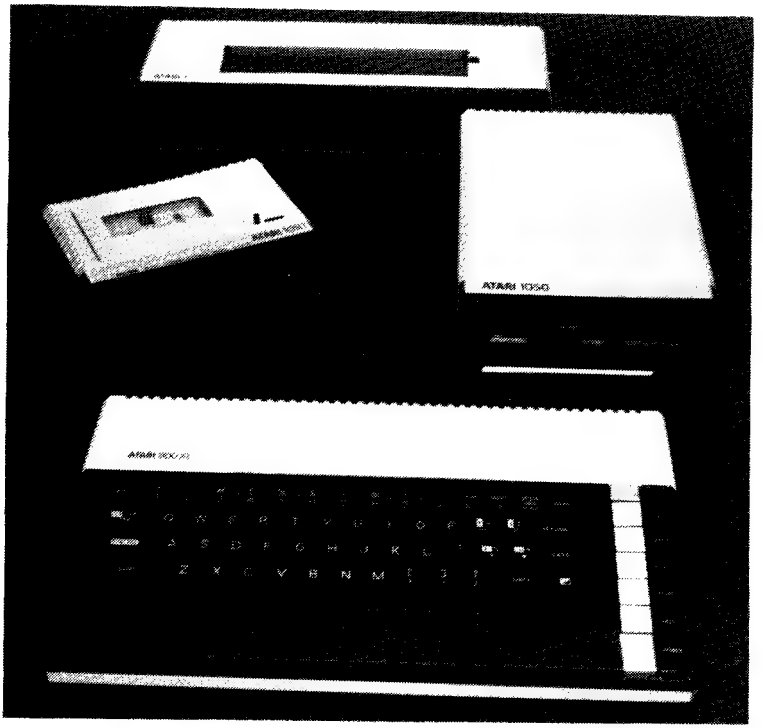
Important Lines in Music Creator

50-135	Initialization
140	Gets input
150-160	Checks to see if user wants to play song or end program
170-310	Finds appropriate data
400	Gets period
500-560	Gets note length
600-620	Plays note just input
700-720	Goes to get next note from user
1000-2020	Plays song
5000-5110	Data for notes, octave, and period value

USING SOUNDS AND MUSIC

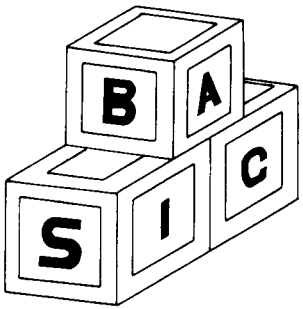
With four voices available on the ATARI, this computer certainly is capable of producing complex pieces of music and interesting, unusual sounds for your programs. Add as many sounds as you can to your games and utilize songs and sounds

in educational programs. You'll probably find that creating these is one of the most entertaining aspects of programming your ATARI computer.



The ATARI 800XL computer system (courtesy Atari, Inc.)

Chapter 3



Educational Software

Computers are often said to be one of the best tools available for education. This may well be true, because computers are the most patient teachers you could ever find. They can remember large amounts of data, recite it accurately, and accept input from any number of students as many times as needed. Computers don't have some of the "human" qualities that real teachers do, but they are excellent at the kinds of work which are repetitive and probably boring for people. Moreover, the student can continue to work with a program until he or she memorizes the information presented by the computer.

In this chapter we'll be looking at four different programs which focus on education. You can easily expand and modify these programs for your own use and for the academic level for the people using the programs. However, these programs are fully functional as they now exist; you are sure to learn something from using these programs—not only about history or French or geography or spelling, but also about programming your ATARI home computer.

HISTORY QUIZ

This program utilizes a large quantity of data by randomly selecting several pieces of information which relate to history. Then it asks you which one particular part of history is usually related to another part. With this program, the computer can

teach you more about history by quizzing you on a variety of subjects and letting you learn by trial and error.

The first part of the program sets up the variables and clears the screen.

```
10   DIM A$(30):DIM B$(30)
20   DIM C$(30):DIM D$(30)
30   DIM E$(1):DIM F$(1)
40   DIM ANSWER$(30)
50   SC=0:TIMES=0
100  GRAPHICS 0
```

The ATARI waits until you are ready to begin and then clears the screen once more:

```
110  POSITION 12,4
120  PRINT"HISTORY QUIZ"
130  POSITION 7,8
140  PRINT"HIT ANY KEY TO CONTINUE"
150  INPUT E$
200  GRAPHICS 0
```

At this point the computer reads the data from the end of the program and selects a random number from 1 to 6. This random number will determine the order in which the aspects of history will be printed—so you'll never know which person or event is associated with another person or event until you are familiar with the facts in this program.

```
205  TIMES=TIMES+1
210  READ A$,B$,C$,D$
220  POSITION 7,4
230  PRINT A$
240  POSITION 7,6
250  PRINT"IS MOST ASSOCIATED WITH:"
300  RAN=INT(6*RND(1)+1)
305  POSITION 2,10
```

The computer then prints out the data and goes to the routine at line 700, which will accept your input.

```
310  IF RAN=1 THEN 400
320  IF RAN=2 THEN 450
330  IF RAN=3 THEN 500
340  IF RAN=4 THEN 550
350  IF RAN=5 THEN 600
360  IF RAN=6 THEN 650
400  PRINT B$
410  PRINT C$
420  PRINT D$
430  GOTO 700
450  PRINT C$
460  PRINT B$
470  PRINT D$
480  GOTO 700
500  PRINT D$
510  PRINT C$
520  PRINT B$
530  GOTO 700
550  PRINT B$
560  PRINT D$
570  PRINT C$
580  GOTO 780
600  PRINT C$
610  PRINT D$
620  PRINT B$
630  GOTO 700
650  PRINT D$
660  PRINT B$
```

```
670  PRINT C$
680  GOTO 700
```

Here the computer accepts your input and checks to see if it is correct or not. Then the ATARI asks if you would like to continue with another question.

```
700  POSITION 7,15
710  INPUT ANSWER$
720  IF ANSWER$<>B$ THEN 800
730  PRINT:PRINT"CORRECT"
740  SC=SC+1
750  GOTO 900
800  PRINT:PRINT"INCORRECT - THE RIGHT ANSWER IS ";B$
810  GOTO 900
900  IF TIMES=22 THEN 2000
910  PRINT"DO YOU WANT TO CONTINUE?"
920  INPUT F$
930  IF F$="Y" THEN 200
```

Finally, here are the data for the program. Remember not to put any extra spaces in the data, since that might make the program malfunction.

```
1000 DATA HERODOTUS,FATHER OF HISTORY,TROJAN WAR,ATTACK
ON ROME
1010 DATA WATERLOO,NAPOLEON,LORD NELSON,DUKE OF WELLINGTON
1020 DATA ANTIETAM,CIVIL WAR,GREECE,WORLD WAR II
1030 DATA HOMER,POETRY,SUBMARINE,JET PLANE
1040 DATA HENRY FORD,INDUSTRIALIST,PRESIDENT,FAMOUS
HISTORIAN
1050 DATA HANNIBAL,ELEPHANTS,BABYLON,PEOPLE EATERS
1060 DATA HAMMURABI,BABYLON,ELEPHANTS,HARDWARE
1070 DATA KING JOHN,MAGNA CARTA,ENGLISH CIVIL WAR,
FOUGHT DUKE OF NORMANDY
```

```
1080 DATA KIT CARSON,WEST,NORTH,SOUTH
1090 DATA SHERMAN,NORTH,WEST,SOUTH
1100 DATA JEB STUART,SOUTH,NORTH,WEST
1110 DATA SAMUEL ADAMS,AMERICAN REVOLUTION,FAMOUS
ECONOMIST,SECOND PRESIDENT
1120 DATA GIBRALTAR,LORD NELSON,DUKE OF WELLINGTON,
NAPOLEON
1130 DATA U.S. SENATE,TRUMAN,JACKSON,EISENHOWER
1140 DATA THOMAS EDISON,MOTION PICTURES,TYPEWRITER,
AIRPLANES
1150 DATA PRINTING,GUTENBERG,LINDERBERG,HEGEL
1160 DATA WILLIAM THE CONQUERER,1066,ROME,1512
1170 DATA KARL MARX,HEGEL,ADAM SMITH,DARWIN
1180 DATA CARTHAGE,ROME,TURKEY,ENGLAND
1190 DATA RENOIR,PAINTER,PHILOSOPHER,INDUSTRIALIST
1200 DATA SHAKESPEARE,ELIZABETH I,KING JOHN,HENRY VIII
1210 DATA JOHN PAUL JONES,AMERICAN REVOLUTION,WAR OF
1812,POET
2000 PRINT:PRINT"YOUR SCORE IS ";INT((SC/TIMES)*100);"
PERCENT."
```

Line 2000, shown above, is the final line in the program. It computes your score by dividing the number of your correct answers by the total number of questions asked, and then multiplying that fraction by 100 to get the percentage.

Important Variables in History Quiz

A$	First word
B$,C$,D$	Words with which to associate the first word
E$, F$	Input needed to continue
ANSWER$	Player's response for quiz
SC	Score
TIMES	Number of questions asked
RAN	Random number

Important Lines in History Quiz

10-50	Initialization
100-150	Introduction
200-250	Beginning of quiz
300-680	Sets up three words in random order
700-810	Gets responses and sees if it's correct
900-930	Sees if player wants to continue
1000-1210	Data
2000	Computes player's score

SPELLING TESTER

If you have trouble with your spelling, as most of us do, or you know someone who could use help, here is a program which will display incorrect and correct spellings of words as shown in Fig. 3-1. You decide which is correct.

The program begins by DIMensioning the variables and clearing the screen:

```
10    DIM A$(15):DIM B$(15)

20    DIM INP$(15):DIM YN$(1)

30    SC=0
```

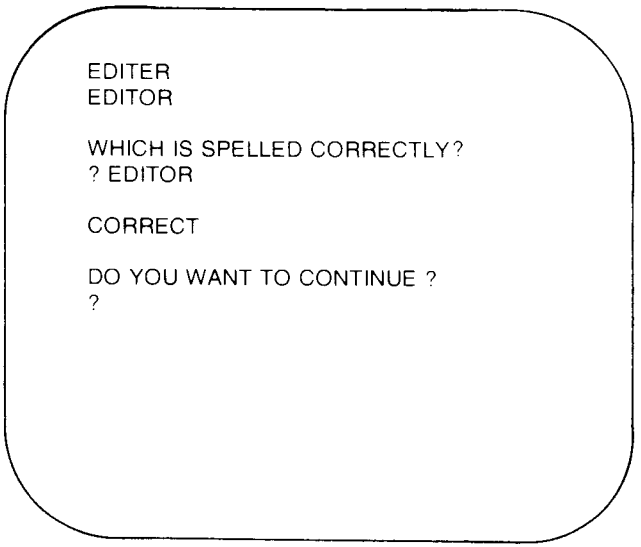


Fig. 3-1. Sample screen for Spelling Checker.

Your ATARI then begins reading the words, displaying the incorrectly and correctly spelled versions in a random order, asking you which one is spelled correctly.

```
100  FOR LOOP1=1 TO 37
110  GRAPHICS 0
120  READ A$,B$
130  RAN=INT(2*RND(1)+1)
140  IF RAN=1 THEN 200
150  PRINT:PRINT:PRINT A$
160  PRINT B$
170  GOTO 300
200  PRINT:PRINT:PRINT B$
210  PRINT A$
300  PRINT:PRINT"WHICH IS SPELLED CORRECTLY"
310  INPUT INP$
```

Once you give an answer, the computer checks to see if you are right or wrong, and also asks if you would like to continue using the program. If not, it will tell you what your score is.

```
320  IF INP$=A$ THEN SC=SC+1:PRINT:PRINT"CORRECT.":
GOTO 500
330  PRINT:PRINT"WRONG."
500  PRINT:PRINT"DO YOU WANT TO CONTINUE?"
510  INPUT YN$
520  IF YN$="N" THEN 600
530  NEXT LOOP1
600  PRINT:PRINT"YOUR SCORE IS ";INT((SC/LOOP1)*100);"
PERCENT."
```

Here, at the end of the program, are the data:

```
2000 DATA IMPERIL,IMPERILL,SHAPELY,SHAPLY,SENSORY,
SENSERY
```

```
2010 DATA HARASS,HARRASS,IMMINENCE,EMMINENCE,IMPALPABLE,
IMPALPIBLE
2020 DATA IMPELLING,IMPELING,PERSISTENT,PERSISTANT,
PROFESSOR,PROFESOR
2030 DATA SUCCEED,SUCCEDE,TOMOROW,TOMMOROW,DISTILL,
DISTIL
2040 DATA BLURRED,BLURED,BOGUS,BOGOS,AROUSE,ARROUSE
2050 DATA HIJACKER,HIGH-JACKER,HARD-CORE,HARDCORE,
HALFBACK,  HALF-BACK
2060 DATA GRITTY,GRITY,GUARANTEE,GUARRANTEE,GROVEL,
GROVELL
2070 DATA ENSUE,INSUE,HIGHLIGHT,HIGH-LIGHT,FOSSIL,
FOSSLE
2080 DATA MISSILE,MISSLE,FULFILL,FULLFIL,ELECTORATE,
ELLECTORATE
2090 DATA EDITOR,EDITER,ELUDE,ELLUDE,DEFER,DEFFER
2100 DATA CONTROL,CONTROLL,CONTROLLED,CONTROLED,
STOPPING,STOPING
2110 DATA MARSHMALLOW,MARSHMELLOW,HYGIENE,HUGENE
2120 DATA FLOTSAM,FLOTSOM,ADVANTAGEOUS,ADVANTAGOUS
```

You might want to try adding words of your own to the program. Simply put in new DATA statements and adjust the FOR/NEXT loop near the beginning of the program so it can accommodate any extra words.

Important Variables in Spelling Tester

A$	Data for the correct spelling
B$	Data for incorrect spelling
SC	Score correct
LOOP1	Quiz
RAN	Random number (1 or 2)
INP$	User's answer

Important Lines in Spelling Tester

10-30	Initialization
100	Beginning of quiz
120	Reads two words from data
130-210	Prints the words in random order
300-330	Asks for answer, sees if it is correct
500-530	Sees if player wants to continue
600	Computers player's score
2000-2120	Data

FRENCH TUTOR

If you are interested in the French language, or if you'd just like to learn a few words to use at your favorite international restaurant, this program will not only teach you French words but also test you on them. Again, you can add more words to the DATA statements, but you should also alter the program to account for the added words.

As usual, the first part of the program sets up the variables and clears the screen:

```
10   DIM E$(15):DIM F$(15)
20   DIM INP$(15):DIM YN$(1)
30   SC=0
100  GRAPHICS 0
```

Then a "menu" is displayed from which you may select an option. The "parts" are the lessons from which you may learn certain words, and the "test" will quiz you on those words.

```
110  POSITION 10,5
120  PRINT"FRENCH LESSON"
130  POSITION 7,10
140  PRINT"1. PART ONE"
150  POSITION 7,11
160  PRINT"2. PART TWO"
170  POSITION 7,12
180  PRINT"3. PART THREE"
190  POSITION 7,13
```

```
200  PRINT"4. PART FOUR"
210  POSITION 7,14
220  PRINT"5. PART FIVE"
230  POSITION 7,15
240  PRINT"6. PART SIX"
250  POSITION 7,16
260  PRINT"7. TEST"
270  INPUT CH
280  ON CH GOTO 500,300,320,340,360,380,1000
```

Once you have selected an option, the computer will go to a different part of the program so it may either teach you or quiz you on the words:

```
300  CLEAR=13
310  GOTO 400
320  CLEAR=26
330  GOTO 400
340  CLEAR=39
350  GOTO 400
360  CLEAR=52
370  GOTO 400
380  CLEAR=65
400  FOR LOOP1=1 TO CLEAR
410  READ F$,E$
420  NEXT LOOP1
500  GRAPHICS 0
510  FOR LOOP2=1 TO 13
520  READ F$,E$
530  PRINT F$
540  PRINT"          ";E$
550  NEXT LOOP2
```

```
600  PRINT:PRINT:PRINT"HIT ANY KEY TO CONTINUE"
610  INPUT INP$
620  RESTORE
630  GOTO 100
```

If you do "take the test" you will have to put in the French word for each English word, using the dialogue shown in Fig. 3-2. The ATARI will keep track of your score as you go.

```
1000 FOR LOOP3=1 TO 78
1010 GRAPHICS 0
1020 READ F$,E$
1030 PRINT:PRINT E$
1040 PRINT:PRINT"WHAT IS THE FRENCH WORD?"
1050 PRINT"(INCLUDE THE ARTICLE)"
1060 INPUT INP$
1070 IF INP$<>F$ THEN 1100
1080 GOTO 1200
```

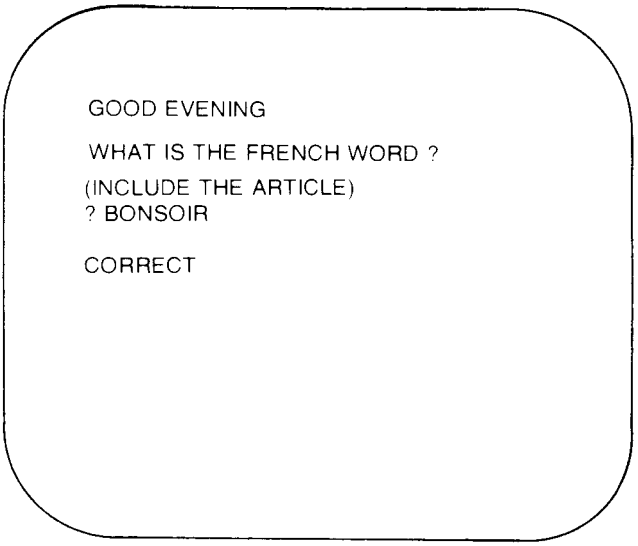


Fig. 3-2. Taking the French test.

```
1100 PRINT:PRINT"WRONG"
1120 GOTO 1500
1200 PRINT:PRINT"CORRECT"
1210 SC=SC+1
```

If you want to continue the program, you may do so when the computer asks you:

```
1500 PRINT:PRINT"DO YOU WANT TO CONTINUE?"
1510 INPUT YN$
1520 IF YN$<>"Y" THEN 1600
1530 NEXT LOOP3
```

Finally, the computer tells you your score based on how well you did on the test.

```
1600 PRINT:PRINT"YOUR SCORE IS ";INT((SC/LOOP3)*100);"
PERCENT."
```

The data are located at the end of the program.

```
2000 DATA L'ADDITION,BILL,L'AGNEAU,LAMB
2010 DATA L'AIDE,ASSISTANCE,L'APRES-MIDI,AFTERNOON
2020 DATA L'AUTOBUS,BUS,LE BAIN,BATH
2030 DATA LE BETE,BEAST,LA BLANCHISSERIE,LAUNDRY
2040 DATA LE BONBON,CANDY,BONJOUR,HELLO
2050 DATA BONSOIR,GOOD EVENING,LA BOULANGERIE,BAKERY
2060 DATA LA BOUTIQUE,SHOP,LA CABINE,CABIN
2070 DATA LE CAMION,TRUCK,LE CHASSEUR,BELLBOY
2080 DATA LE CHAUSSURE,SHOE,LE CHEQUE,CHECK
2090 DATA LE CHIEN,DOG,LE COMPLET,SUIT
2100 DATA LA CRAVATE,NECKTIE,LE CUIT,LEATHER
2110 DATA DE,FROM,DEMAIN,TOMMOROW
2120 DATA LA DOUANE,CUSTOMS,LE DRAPEAU,FLAG
2130 DATA L'ESSENCE,GASOLINE,LA FEMME,WOMAN
```

```
2140 DATA LE GARCON,WAITER,LA GARE,STATION
2150 DATA LE GENDARME,POLICEMAN,LA GLACE,ICE CREAM
2160 DATA GRATUIT,FREE,L'HORAIRE,SCHEDULE
2170 DATA L'HOROLOGE,CLOCK,L'HOTELIER,INNKEEPER
2180 DATA L'IMPOT,TAX,INTERDIT,FORBIDDEN
2190 DATA LUNDI,MONDAY,MARDI,TUESDAY
2200 DATA MERCREDI,WEDNESDAY,JEUDI,THURSDAY
2210 DATA VENDREDI,FRIDAY,SAMEDI,SATURDAY
2220 DATA DIMANCHE,SUNDAY,LE JOUET,TOY
2230 DATA LE JOUR,DAY,LE JOURNAL,NEWSPAPER
2240 DATA LE LIBRAIRIE,BOOKSTORE,LA MAIRIE,TOWN HALL
2250 DATA LE MATIN,MORNING,LE MEDECIN,DOCTOR
2260 DATA MERCI,THANK YOU,LE METRO,SUBWAY
2270 DATA LE MIDI,NOON,LE MINUIT,MIDNIGHT
2280 DATA LE MOIS,MONTH,L'OEUF,EGG
2290 DATA LE PAIN,BREAD,PERDU,LOST
2300 DATA LE PERE,FATHER,LE PLAN,CITY MAP
2310 DATA LE POISSON,FISH,LE PORTEUR,PORTER
2320 DATA LE POULET,CHICKEN,LE QUARTIER,NEIGHBORHOOD
2330 DATA LE REVEIL,ALARM CLOCK,LA ROBE,DRESS
2340 DATA LE SAC,BAG,LA SALLE,ROOM
2350 DATA LA SEMAINE,WEEK,LA SERVEUSE,WAITRESS
2360 DATA LE STYLO,PEN,LA TAILLE,SIZE
2370 DATA LE TAILLEUR,TAILOR,LA VALISE,SUITCASE
2380 DATA LA VIANDE,MEAT,LA VOITURE,CAR
```

If you would like to learn a language other than French, you can use the same technique in this program. Simply include the data for the English and the foreign-language words.

Important Variables in French Tutor

E$	English word in data

F$	French word in data
INP$	Player's response to test
YN$	Used to ask if player wants to continue
SC	Score in quiz
CH	Choice from menu (input)
CLEAR	Number of words and definitions in data to skip over
LOOP1	Used to do the above
LOOP2	Used to read data and print information on screen
LOOP3	Used to ask questions for test

Important Lines in French Tutor

10-30	Initialization
100-260	Sets up menu
270-280	Goes to selected item
300-380	Sets number of pairs of data to skip over
400-420	Does the above
500-550	Reads desired data, prints it out
600-630	Goes back to menu
1000	Beginning of test
1010-1050	Asks question
1060-1080	Gets response
1100-1210	Tells player if he's right or wrong
1500-1530	Asks player if he wants to continue
1600	Computes player's score

STATES AND CAPITALS

At some time in every American's life he or she must learn the states and capitals. As terrible as this task may be, the computer can make it more bearable for you, as you will discover with this program.

First the computer sets up the program and gives you the option of taking a quiz on the states and their capitals, or simply finding information about which state has what capital, or vice versa.

```
5    DIM A$(50):DIM B$(50)

6    DIM INP$(20):DIM USED(50)

7    DIM RESPONSE$(20):DIN YN$(1)

10   GRAPHICS 0
```

```
20   POSITION 9,4
30   PRINT"STATES AND CAPITALS MENU"
40   POSITION 7,7
50   PRINT"1. INFORMATION"
60   POSITION 7,9
70   PRINT"2. QUIZ"
80   POSITION 7,14
90   PRINT"PLEASE CHOOSE ONE."
95   POSITION 7,16
100  INPUT CH
110  IF CH>2 THEN 80
130  IF CH=1 THEN 150
140  IF CH=2 THEN 1000
```

If you want information, you simply tell the computer that you want to find out which state belongs to what capital or which capital belongs to what state. (See Fig. 3-3.) Once you have done this, the computer will find the information.

```
150  GRAPHICS 0
160  POSITION 7,7
```

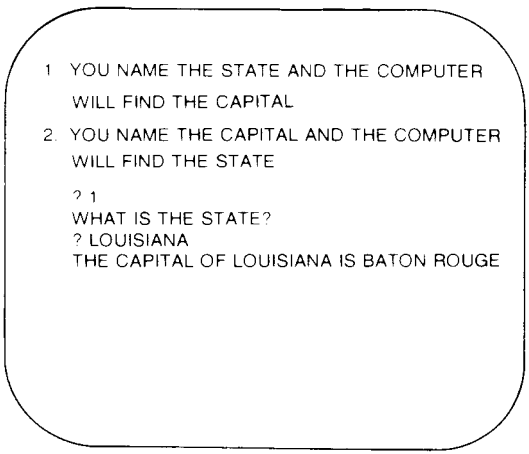


Fig. 3-3. Getting information from States and Capitals.

```
170  PRINT"1. YOU NAME THE STATE AND THE COMPUTER WILL
FIND THE CAPITAL."
180  POSITION 7,10
190  PRINT"2. YOU NAME THE CAPITAL AND THE COMPUTER WILL
FIND THE STATE."
200  POSITION 7,15
205  PRINT"PLEASE INPUT YOUR CHOICE";
210  INPUT CH2
220  ON CH2 GOTO 240,400
230  GOTO 200
240  GRAPHICS 0
250  POSITION 7,7
260  PRINT"WHAT IS THE STATE?"
270  INPUT INP$
280  FOR L1=1 TO 50
290  READ A$,B$
300  IF A$=INP$ THEN 320
310  NEXT L1
315  IF L1=51 THEN PRINT"INVALID INPUT":GOTO 260
320  POSITION 7,11
330  PRINT"THE CAPITAL OF ";A$;" IS ";B$
340  FOR DELAY=1 TO 750:NEXT DELAY
345  RESTORE
350  GOTO 10
400  GRAPHICS 0
410  POSITION 7,7
420  PRINT"WHAT IS THE CAPITAL?
430  INPUT INP$
440  FOR L1=1 TO 50
450  READ A$,B$
```

```
460  IF B$=INP$ THEN 480
470  NEXT L1
475  IF L1=51 THEN PRINT"INVALID INPUT":GOTO 420
480  POSITION 7,11
490  PRINT B$;" IS THE CAPITAL OF ";A$
500  GOTO 340
```

Here are the data of the states and their capitals required by this program:

```
800  DATA ALABAMA,MONTGOMERY,ALASKA,JUNEAU
810  DATA ARIZONA,PHOENIX,ARKANSAS,LITTLE ROCK
820  DATA CALIFORNIA,SACRAMENTO,COLORADO,DENVER
830  DATA CONNECTICUT,HARTFORD,DELAWARE,DOVER
840  DATA FLORIDA,TALLAHASEE,GEORGIA,ATLANTA
850  DATA HAWAII,HONOLULU,IDAHO,BOISE
860  DATA ILLINOIS,SPRINGFIELD,INDIANA,INDIANAPOLIS
870  DATA IOWA,DES MOINES,KANSAS,TOPEKA
880  DATA KENTUCKY,FRANKFORT,LOUISIANA,BATON ROUGE
890  DATA MAINE,AUGUSTA,MARYLAND,ANNAPOLIS
900  DATA MASSACHUSETTS,BOSTON,MICHIGAN,LANSING
910  DATA MINNESOTA,ST. PAUL,MISSISSIPPI,JACKSON
920  DATA MISSOURI,JEFFERSON CITY,MONTANA,HELENA
930  DATA NEBRASKA,LINCOLN,NEVADA,CARSON CITY
940  DATA NEW HAMPSHIRE,CONCORD,NEW JERSEY,TRENTON
950  DATA NEW MEXICO,SANTA FE,NEW YORK,ALBANY
960  DATA NORTH CAROLINA,RALEIGH,NORTH DAKOTA,BISMARK
970  DATA OHIO,COLUMBUS,OKLAHOMA,OKLAHOMA CITY
980  DATA OREGON,SALEM,PENNSYLVANIA,PHILADELPHIA
990  DATA RHODE ISLAND,PROVIDENCE,SOUTH CAROLINDA,
COLUMBIA
992  DATA SOUTH DAKOTA,PIERRE,TENNESSEE,NASHVILLE
993  DATA TEXAS,AUSTIN,UTAH,SALT LAKE CITY
```

```
995  DATA VERMONT,MONTPELIER,VIRGINIA,RICHMOND
997  DATA WASHINGTON,OLYMPIA,WEST VIRGINIA,CHARLESTON
998  DATA WISCONSIN,MADISON,WYOMING,CHEYENNE
```

And, finally, if you want to be tested by the computer, this part of the program will require you to name either the state of a capital or the capital of a state, whichever you prefer.

```
1000 GRAPHICS 0
1003 FOR L5=1 TO 50
1005 USED(L5)=0
1007 NEXT L5
1010 POSITION 7,7
1020 PRINT"1. 'WHAT IS THE CAPITAL?'"
1030 POSITION 7,9
1040 PRINT"2. 'WHAT IS THE STATE?'"
1050 POSITION 7,12
1060 INPUT CH3
1070 IF CH3=1 THEN 1100
1080 IF CH3=2 THEN 1100
1090 GOTO 1060
1100 GRAPHICS 0
1105 FOR L2=1 TO 50
1110 RAN=(50*RND(1)+1)
1120 FOR L3=1 TO 50
1130 IF RAN=USED(L3) THEN 1110
1140 NEXT L3
1150 USED(L2)=RAN
1160 FOR L4=1 TO RAN
1170 READ A$,B$
1180 NEXT L4
1190 RESTORE
```

```
1200 IF CH3=1 THEN 1300
1210 IF CH3=2 THEN 1400
1300 PRINT"WHAT IS THE CAPITAL OF ";A$;"?"
1310 INPUT RESPONSE$
1320 IF RESPONSE$=B$ THEN PRINT"CORRECT":SC=SC+1:
GOTO 1500
1330 PRINT"WRONG"
1340 PRINT"THE CAPITAL OF ";A$;" IS ";B$
1350 GOTO 1500
1400 PRINT B$;" IS THE CAPITAL OF WHAT STATE?"
1410 INPUT RESPONSE$
1420 IF RESPONSE$=A$ THEN PRINT"CORRECT":SC=SC+1:
GOTO 1500
1430 PRINT"WRONG"
1440 PRINT B$;" IS THE CAPITAL OF ";A$
1450 GOTO 1500
```

You can keep answering questions until you are satisfied that you know the material. Then the computer will give you your score.

```
1500 PRINT"DO YOU WANT TO TRY ANOTHER?"
1510 INPUT YN$
1520 IF YN$="Y" THEN 1600
1525 GRAPHICS 0
1530 NEXT L2
1600 PRINT"YOUR SCORE IS ";INT(SC/L2)*100;" PERCENT."
```

You could use this same program concept to match any two things, such as international capitals with their respective countries, of scientific achievements with people who made them possible, and so on. There are many possibilities in educational software, and you can probably come up with an excellent subject that a person could learn with the help of the ATARI.

Important Variables in States and Capitals

A$	Data for the state
B$	Data for capital
INP$	Input for finding information
USED(X)	Makes sure that states in quiz are not used twice
RESPONSE$	Player's answer in quiz
YN$	Sees if player wants to continue
CH	Player's choice from menu
CH2	Choice of information wanted
CH3	Choice of type of quiz wanted
L1	Loop for finding information
DELAY	Delay loop
L2	Loop to ask quiz questions
L3	Loop used to make sure that same state has not been used twice in quiz
L4	Used to read remaining data
L5	Sets all USED(X)=0

Important Lines in States and Capitals

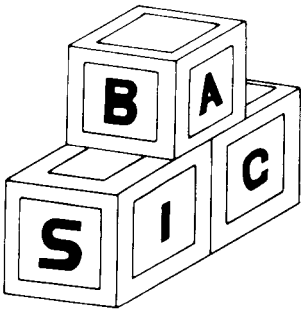
5-7	Initialization
10-95	Sets up menu
100-140	Gets user's choice
150-230	Gets choice of information wanted
240-350	Finds capital of a state
400-500	Finds the state
800-998	Data
1003-1007	Sets all USED(X) to zero
1010-1090	Gets choice of quiz
1100	Beginning of quiz
1120-1140	Makes sure that random number has not been used twice
1160-1180	Reads data to get to the data needed in the quiz
1300-1450	Asks questions, gets responses, sees if they are correct
1500-1520	Sees if user wants to continue the quiz
1600	Computes the user's score

LEARNING THE EASY WAY

As you can tell from using these programs, computers can make learning more enjoyable and less frightening for younger

people, who might be afraid of being scorned for their mistakes. Computers can teach individuals at their own pace, and will provide endless encouragement and congratulations for people learning with the computer. See if you can make a program to teach a subject in which you have an interest, and find out if people using the program can learn as much or more than they would with a teacher. You might be surprised at the results.

Chapter 4



Math and Your Computer

Since computers are basically "math machines" (they work only with numbers and translate words into numbers before using them), your ATARI 600XL or 800XL is very proficient at math. It can handle addition, subtraction, multiplication, division, trigonometric functions, exponents, and just about any other mathematical formula you can give it.

Because of the ATARI's prowess at mathematics, your computer can be an excellent tool for solving your math problems, as well as testing you on your math abilities. In this chapter we'll take a look at four programs that serve these purposes and will prove how quickly and accurately the ATARI home computers can handle numbers.

CALCULATOR

This first program will serve the functions of an advanced calculator. It will perform the four basic functions (add, subtract, multiply, divide), trigonometric functions (like sine, cosine, tangent), and find the square root of a number. This program takes advantage of most of the ATARI's math commands and can help you by being a powerful "computer calculator," as shown in Fig. 4-1.

The first part of the program clears the screen and displays the "menu" of options. The computer then allows you to choose a function and input the number you wish to use with it.

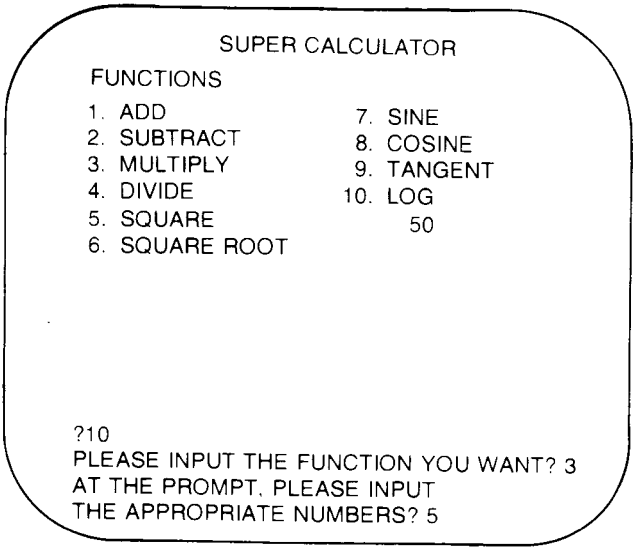


Fig. 4-1. Computer Calculator.

```
10 GRAPHICS 0
20   DEG
100  POSITION 9,1
110  PRINT"SUPER CALCULATOR"
120  PRINT
130  PRINT"FUNCTIONS:"
140  PRINT
150  PRINT"1. ADD          7. SINE"
160  PRINT"2. SUBTRACT     8. COSINE"
170  PRINT"3. MULTIPLY     9. TANGENT"
180  PRINT"4. DIVIDE      10. LOG"
190  PRINT"5. SQUARE"
195  PRINT"6. SQUARE ROOT"
210  POSITION 5,15
220  PRINT NM;"                "
225  PRINT:PRINT
```

```
230  INPUT NM
240  PRINT"PLEASE INPUT THE FUNCTION YOU WANT";
250  INPUT FUNC:PRINT"AT THE PROMPT SIGN, PLEASE INPUT"
251  PRINT"THE APPROPRIATE NUMBERS"
```

The computer then goes to a predetermined line number based on the value of your function. If the function number you selected does not exist, the computer will go back to line 210 to ask for your input once again.

```
300  ON FUNC GOTO 500,600,700,800,900,1000,1100,1200,
1300,1400
310  GOTO 210
```

For some of these functions (like square root), only one number (variable NM) will be necessary. Other functions require a second number (variable NM2). In the lines below, you will see that some of the functions INPUT to the variable NM2, while others immediately work with the variable NM and return to the functions menu. The result is displayed and the computer waits for more input.

```
500  INPUT NM2
510  NM=NM+NM2
520  GOTO 210
600  INPUT NM2
610  NM=NM-NM2
620  GOTO 210
700  INPUT NM2
710  NM=NM*NM2
720  GOTO 210
800  INPUT NM2
810  NM=NM/NM2
820  GOTO 210
900  NM=NM*NM
910  GOTO 210
1000 NM=SQR(NM)
```

```
1010 GOTO 210

1100 NM=SIN(NM)

1110 GOTO 210

1200 NM=COS(NM)

1210 GOTO 210

1300 NM=SIN(NM)/COS(NM)

1310 GOTO 210

1400 NM=CLOG(NM)

1410 GOTO 210

1500 NM=INT(NM)

1510 GOTO 210
```

In all of the functions above, the computer returns to line 210 to display the result of the function and await input for another problem. You might want to add more functions to this program using your own special equations. If you wish to do this, simply add lines to the end of the program (your first added function might be on lines 1600 and 1610) and add the line number(s) to the ON . . . GOTO statement in line 300.

Important Variables in Calculator

NM	The number that is calculated
FUNC	The function that is chosen
NM2	Second number for adding, subtracting, or whatever

Important Lines in Calculator

20	Sets trigonometric functions to degrees
100-200	Sets up list of functions
220	Displays number
225-250	Gets number and function by which to calculate
300	Goes to chosen function

BASIC SKILLS CHECK-UP

If you think that calculators and computers have "spoiled" you into complete reliance on machines for simple mathematical functions, you might want to try this program. Or you could let a younger person use it to try out his or her skills at addition, subtraction, and multiplication. The challenge of this program is

not only getting the problems correct, but also doing so in as little time as possible.

The program begins by setting up the string arrays. It then waits for you to hit a key and press <RETURN> so it can begin.

```
50   DIM A$(1)

60   DIM YN$(1)

100  GRAPHICS 0

110  POSITION 7,8

120  PRINT"BASIC SKILLS CHECK-UP"

130  POSITION 4,12

140  PRINT"HIT ANY KEY TO CONTINUE"

150  INPUT A$
```

The ATARI now clears the screen and shows you the three options from which you may choose. Select one of these, and press <RETURN>.

```
160  GRAPHICS 0

170  POSITION 5,8

180  PRINT"1. ADDITION"

185  POSITION 5,9

190  PRINT"2. SUBTRACTION"

195  POSITION 5,10

200  PRINT"3. MULTIPLICATION"

210  PRINT:INPUT CH
```

Because the computer has to set a certain limit how high the random numbers can be used to test you, the ATARI takes into consideration the function you have chosen. If you are going to be doing addition or subtraction, the limit of the random numbers will be from one to fifteen, since these are not too high for a person adding or subtracting. On the other hand, since multiplication is more difficult, the limit is set to nine. Also, the timer in the ATARI (in bytes 19 and 20 in the computer's memory) is set to zero. The computer then begins the FOR/NEXT loop to select ten random numbers for your problems.

```
215  GRAPHICS 0
220  LIMIT=15
230  IF CH=3 THEN LIMIT=9
240  POKE 19,0:POKE 20,0
250  FOR TIMES=1 TO 10
270  NUM1=INT(LIMIT*RND(1)+1)
280  NUM2=INT(LIMIT*RND(1)+1)
```

Based on your selection, the computer will present a problem to you and accept an answer.

```
290  ON CH GOTO 300,320,340
300  PRINT NUM1;"+";NUM2;"=";
310  INPUT ANSWER:GOTO 500
320  PRINT NUM1;"-";NUM2;"=";
330  INPUT ANSWER:GOTO 500
340  PRINT NUM1;"*";NUM2;"=";
350  INPUT ANSWER:GOTO 500
```

Once you have selected an answer the computer will check to see if you are right or wrong; if you are correct, it will go to the next question or repeat the question if you answered incorrectly.

This program is an excellent tool for sharpening up basic math skills since you have to work against the clock to answer the questions. To make it more difficult for the person using the program (even if that happens to be you), try to limit the amount of time in which questions can be answered. You could also increase the high limit on the random numbers that can be selected.

Important Variables in Basic Skills Check-Up

A$	Input to see if user is ready to continue
YN$	Input to see if user wants to use the program again
CH	User's choice of function
LIMIT	Sets limit of random numbers to 15 when doing addition or subtraction and to 9 when doing multiplication

NUM1	First random number
NUM2	Second random number
A	The time the user takes to complete the problems
ANSWER	The user's answer
TIMES	The loop which asks for the answers to ten problems
SECONDS	User's time (variable A) converted to seconds

Important Lines in Basic Skills Check-Up

50-210	Initialization to set up menu
220-230	Sets upper limit of random numbers
240	Sets time to zero
250-280	Gets two random numbers
290-350	Prints selected random operation on screen and gets answer from user
500-560	Checks if answer is right or wrong
700	Tells user he is wrong, goes to ask question once again
800	Tells user he's right
900-920	Gets time, converts it to seconds
930-940	Tells user his time
950-980	Asks if the user wants to try again

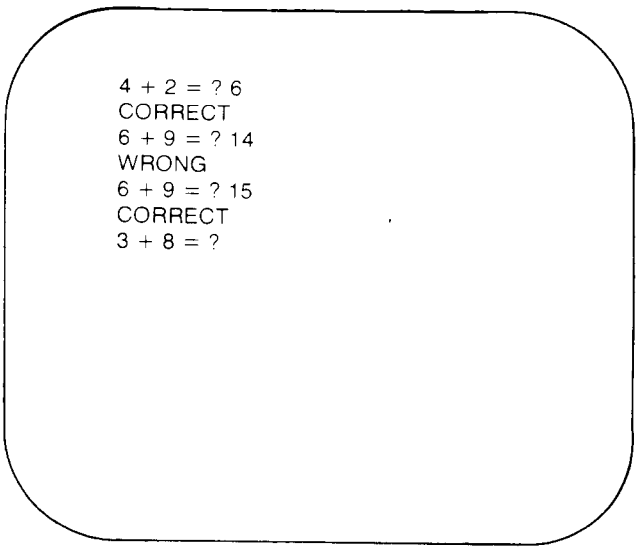


Fig. 4-2. Skills Check-Up.

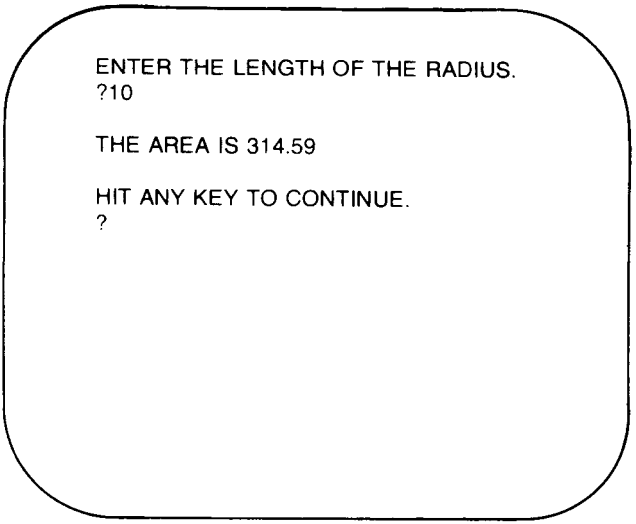


Fig. 4-3. Geometry program finding the area of a circle.

GEOMETRY

This next program is basically a tool, since it will solve geometry problems for you. With this tool you can find areas and volumes; you can also discover how the computer finds the answer to these problems. Figure 4-3 shows a sample problem. To use Geometry, you simply select a particular function and then give the computer the necessary data to solve the problem.

```
500  ON CH GOTO 510,530,550
510  IF ANSWER<>(NUM1+NUM2) THEN GOTO 700
520  GOTO 800
530  IF ANSWER<>(NUM1-NUM2) THEN GOTO 700
540  GOTO 800
550  IF ANSWER<>(NUM1*NUM2) THEN GOTO 700
560  GOTO 800
700  PRINT"WRONG":GOTO 290
800  PRINT"CORRECT":NEXT TIMES
```

Once the ten math questions have been asked, the ATARI will inform you of the time it took you to answer the problems, and

then ask if you would like to use the program again. If so, you can select another function to practice. If not, the program will end.

```
900  A=PEEK(19)
910  SECONDS=4.27*A
920  SECONDS=INT(SECONDS)
930  PRINT:PRINT
940  PRINT"YOUR TIME IS ";SECONDS;" SECONDS."
950  PRINT:PRINT
960  PRINT"DO YOU WANT TO PLAY AGAIN";
970  INPUT YN$
980  IF YN$="Y" THEN RUN
```

First, the computer shows you the 16 different functions available to you and lets you tell it which function you wish to use.

```
50   DIM ANY$(1)
100  GRAPHICS 0
110  PRINT:PRINT:PRINT
120  PRINT"1. AREA OF A RECTANGLE"
130  PRINT"2. AREA OF A SQUARE"
140  PRINT"3. AREA OF A PARALLELOGRAM"
150  PRINT"4. AREA OF A TRIANGLE"
160  PRINT"5. AREA OF A TRAPEZOID"
170  PRINT"6. AREA OF A CIRCLE"
180  PRINT"7. AREA OF A SPHERE"
190  PRINT"8. AREA OF A CUBE"
200  PRINT"9. AREA OF A CYLINDER"
210  PRINT"10. AREA OF A SPHERE"
220  PRINT"11. VOLUME OF A RECTANGULAR SOLID"
230  PRINT"12. VOLUME OF A CUBE"
240  PRINT"13. VOLUME OF A CYLINDER"
250  PRINT"14. VOLUME OF A SPHERE"
260  PRINT"15. VOLUME OF A CONE"
```

```
270  PRINT"16. VOLUME OF A PYRAMID"
280  PRINT:PRINT:PRINT"PLEASE CHOOSE ONE."
290  INPUT CHOICE
```

Your ATARI will now go to the line number that performs the function selected.

```
295  GRAPHICS 0
300  ON CHOICE GOTO 350,400,500,600,700,800,900,1000,
1100,1200,1300,1400,1500,1600,1700,1800
```

Each of the formulae below has to know certain facts to calculate the answer to your problem. For instance, to find the area of a square, the computer needs only the length of one side. To find the volume of a cone, however, the computer needs to find the radius of the base and the height of the cone. It finds out this data by accessing certain subroutines found near the end of the program. These subroutines will let you input the necessary data. Because the computer can use them over and over again, this program is shorter and more efficient than it would be if, for each of the problems, the computer had INPUT statements exclusively for that type of function. In the lines below, the computer will access the subroutines needed to get the necessary data, and then perform the function, go to the appropriate line number, and displays the result.

```
350  GOSUB 2000
360  GOSUB 2100
370  AREA=LN*WD
380  GOTO 6000
400  GOSUB 2000
410  AREA=LN*LN
420  GOTO 6000
500  GOSUB 2200
510  GOSUB 2300
520  AREA=BA*HT
530  GOTO 6000
```

```
600  GOSUB 2200
610  GOSUB 2300
620  AREA=(BA*HT)/2
630  GOTO 6000
700  GOSUB 2200
710  GOSUB 2400
720  GOSUB 2300
730  AREA=((BA+BA2)/2)*HT
740  GOTO 6000
800  GOSUB 2500
810  AREA=(RD*RD)*3.14159
820  GOTO 6000
900  GOSUB 2000
910  GOSUB 2100
920  GOSUB 2300
930  AREA=(2*LN*WD)+(4*LN*HT)
940  GOTO 6000
1000 GOSUB 2000
1010 AREA=6*LN*LN
1020 GOTO 6000
1100 GOSUB 2500
1110 GOSUB 2300
1120 AREA=2*RD*HT*3.14159
1130 GOTO 6000
1200 GOSUB 2500
1210 AREA=4*RD*RD*3.14159
1220 GOTO 6000
1300 GOSUB 2000
1310 GOSUB 2100
1320 GOSUB 2300
1330 VOL=LN*WD*HT
```

```
1340 GOTO 6500
1400 GOSUB 2000
1410 VOL=LN*LN*LN
1420 GOTO 6500
1500 GOSUB 2300
1510 GOSUB 2500
1520 VOL=3.14159*RD*RD*HT
1530 GOTO 6500
1600 GOSUB 2500
1610 VOL=(4*3.14159*RD*RD*RD)/3
1620 GOTO 6500
1700 GOSUB 2300
1710 GOSUB 2500
1720 VOL=(3.14159*RD*RD*HT)/3
1730 GOTO 6500
1800 GOSUB 2200
1810 GOSUB 2300
1820 VOL=(BA*BA*HT)/3
1830 GOTO 6500
```

The lines below are the subroutines mentioned earlier. They allow you to enter the width, height, radius, or whatever else is needed to calculate an accurate answer about a geometric shape or object.

```
2000 PRINT:PRINT"ENTER THE LENGTH."
2010 INPUT LN
2020 RETURN
2100 PRINT:PRINT"ENTER THE WIDTH."
2110 INPUT WD
2120 RETURN
2200 PRINT:PRINT"ENTER THE LENGTH OF THE BASE."
```

```
2210 INPUT BA
2220 RETURN
2300 PRINT:PRINT"ENTER THE HEIGHT."
2310 INPUT HT
2320 RETURN
2400 PRINT:PRINT"ENTER THE LENGTH OF THE SECOND BASE."
2410 INPUT BA2
2420 RETURN
2500 PRINT:PRINT"ENTER THE LENGTH OF THE RADIUS."
2510 INPUT RD
2520 RETURN
```

Once the ATARI has found the answer to your problem, it will display the area or the volume, whichever is appropriate.

```
6000 PRINT:PRINT
6010 PRINT"THE AREA IS ";AREA
6020 GOTO 7000
6500 PRINT:PRINT
6510 PRINT"THE VOLUME IS ";VOL
```

The ATARI will then wait for you to press a key followed by the <RETURN> key to try another problem.

```
7000 PRINT:PRINT
7010 PRINT"HIT ANY KEY TO CONTINUE."
7020 INPUT ANY$
7030 GOTO 100
```

Again, you can put other formulae into this program, but you will have to compensate for the additions in the other parts of your program (such as the ON . . . GOTO statement). Incidentally, not only is this an excellent tool for calculating the answers to problems, but it could also be used (by a high school teacher, for example) to check geometry tests.

Important Variables in Geometry

CHOICE	Input for the figure chosen by the user
AREA	Area of a figure
VOL	Volume of a figure
LN	Length
WD	Width
HT	Height
BA	Base length
BA2	Second base length
RD	Radius
ANY$	Used to see if user wants to continue

Important Line Numbers in Geometry

50-290	Sets up menu
300	Goes to selected figure
350-1830	Calculates areas or volume
2000-2520	Gets necessary input from user
6000-6020	Tells user the area
6500-6510	Tells user the volume
7000-7030	Waits until user is ready to continue

LENGTH CONVERTER

This last program is also a tool. It can convert linear units of one type, specifically measurements of length in either metric or "standard" units—as nonstandard as the American system is with respect to the rest of the world—into linear units of another type. For example, you could change a number of kilometers into miles, a number of centimeters into inches, or a number of kilometers into meters. You can change from any unit to any other unit, even metric to metric or standard to standard. A sample screen can be found in Fig. 4-4.

As usual, the first part of this program shows a menu of the different selections. You can chose from this menu the number of the unit you will give the computer and the unit to which you want to change. After the computer shows you the menu, it goes to the subroutine at line 4000, which will let you select both units and the quantity involved.

```
50   DIM INP$(1)

100  GRAPHICS 0

200  POSITION 10,3
```

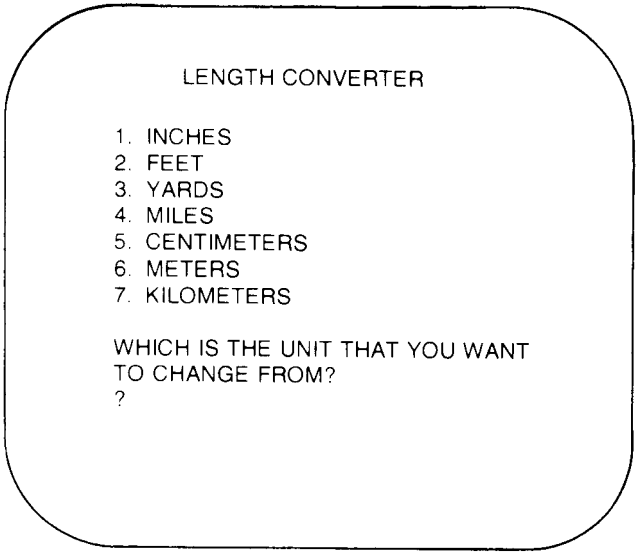


Fig. 4-4. Length Converter.

```
250  PRINT"LENGTH CONVERTER"
260  PRINT:PRINT
300  PRINT"1. INCHES"
310  PRINT"2. FEET"
320  PRINT"3. YARDS"
330  PRINT"4. MILES"
340  PRINT"5. CENTIMETERS"
350  PRINT"6. METERS"
360  PRINT"7. KILOMETERS"
370  GOSUB 4000
```

The ATARI will then go to a line number based on the value of the variable UNIT1. At that line number will be another ON . . . GOTO statement which will, based on the value of UNIT2, find the line which will convert your number from the first unit to the second unit amount. At that point, the ATARI will go to line 5000 to give you the answer to the question.

```
380  ON UNIT1 GOTO 400,500,600,700,800,900,1000
400  ON UNIT2 GOTO 410,420,430,440,450,460,470
410  AMOUNT2=AMOUNT1:GOTO 5000
420  AMOUNT2=AMOUNT1/12:GOTO 5000
430  AMOUNT2=AMOUNT1/36:GOTO 5000
440  AMOUNT2=(AMOUNT1/12)/5280:GOTO 5000
450  AMOUNT2=AMOUNT1*2.54:GOTO 5000
460  AMOUNT2=AMOUNT1*.0254:GOTO 5000
470  AMOUNT2=((AMOUNT1/12)/5280)*.62:GOTO 5000
500  ON UNIT2 GOTO 510,520,530,540,550,560,570
510  AMOUNT2=AMOUNT1*12:GOTO 5000
520  AMOUNT2=AMOUNT1:GOTO 5000
530  AMOUNT2=AMOUNT1/3:GOTO 5000
540  AMOUNT2=AMOUNT1/5280:GOTO 5000
550  AMOUNT2=(AMOUNT1*12)*2.54:GOTO 5000
560  AMOUNT2=AMOUNT1*0.3048:GOTO 5000
570  AMOUNT2=(AMOUNT1/5280)*1.61:GOTO 5000
600  ON UNIT2 GOTO 610,620,630,640,650,660,670
610  AMOUNT2=AMOUNT1*24:GOTO 5000
620  AMOUNT2=AMOUNT1*3:GOTO 5000
630  AMOUNT2=AMOUNT1:GOTO 5000
640  AMOUNT2=(AMOUNT1*3)/5280:GOTO 5000
650  AMOUNT2=(AMOUNT1*24)*2.54:GOTO 5000
660  AMOUNT2=AMOUNT1*0.91:GOTO 5000
670  AMOUNT2=(AMOUNT1/1760)*1.61:GOTO 5000
700  ON UNIT2 GOTO 710,720,730,740,750,760,770
710  AMOUNT2=(AMOUNT1*5280)*12:GOTO 5000
720  AMOUNT2=AMOUNT1*5280:GOTO 5000
730  AMOUNT2=AMOUNT1*1760:GOTO 5000
740  AMOUNT2=AMOUNT1:GOTO 5000
```

```
750  AMOUNT2=(AMOUNT1*63360)*2.54:GOTO 5000
760  AMOUNT2=(AMOUNT1*1.61)*1000:GOTO 5000
770  AMOUNT2=AMOUNT1*1.61:GOTO 5000
800  ON UNIT2 GOTO 810,820,830,840,850,860,870
810  AMOUNT2=AMOUNT1*0.39:GOTO 5000
820  AMOUNT2=(AMOUNT1*0.39)/12:GOTO 5000
830  AMOUNT2=((AMOUNT1*0.39)/12)/3:GOTO 5000
840  AMOUNT2=((AMOUNT1*0.39/12))*5280:GOTO 5000
850  AMOUNT2=AMOUNT1:GOTO 5000
860  AMOUNT2=AMOUNT1/100:GOTO 5000
870  AMOUNT2=AMOUNT1/1000:GOTO 5000
900  ON UNIT2 GOTO 910,920,930,940,950,960,970
910  AMOUNT2=AMOUNT1/0.254:GOTO 5000
920  AMOUNT2=(AMOUNT1/0.0254)/12:GOTO 5000
930  AMOUNT2=AMOUNT1*1.09:GOTO 5000
940  AMOUNT2=((AMOUNT1/.0254)/12)*5280)*12:GOTO 5000
950  AMOUNT2=AMOUNT1*100:GOTO 5000
960  AMOUNT2=AMOUNT1:GOTO 5000
970  AMOUNT2=AMOUNT1/1000:GOTO 5000
1000 ON UNIT2 GOTO 1010,1020,1030,1040,1050,1060,1070
1010 AMOUNT2=((AMOUNT1*.62)*5280)*12:GOTO 5000
1020 AMOUNT2=(AMOUNT1*0.62)*5280:GOTO 5000
1030 AMOUNT2=(AMOUNT1*0.62)*1760:GOTO 5000
1040 AMOUNT2=AMOUNT1*0.62:GOTO 5000
1050 AMOUNT2=AMOUNT1*100000:GOTO 5000
1060 AMOUNT2=AMOUNT1*1000:GOTO 5000
1070 AMOUNT2=AMOUNT1:GOTO 5000
```

The subroutine below will let you input the unit from which you want to change, the amount of the unit, and the unit to which you want to change.

```
4000 PRINT:PRINT"WHICH IS THE UNIT THAT YOU WANT TO
CHANGE FROM?"
4010 INPUT UNIT1
4020 PRINT:PRINT"WHAT IS THE AMOUNT OF THIS UNIT?"
4030 INPUT AMOUNT1
4040 PRINT:PRINT"WHAT UNIT DO YOU WISH TO CHANGE TO?"
4050 INPUT UNIT2
4060 RETURN
```

Finally, the ATARI will tell you the answer in the new unit, and then wait for you to press a key plus <RETURN> to do another conversion.

```
5000 PRINT:PRINT"THE AMOUNT OF THE NEW UNIT IS ";AMOUNT2
5010 PRINT:PRINT"HIT ANY KEY TO CONTINUE."
5020 INPUT INP$
5030 GOTO 100
```

This completes this program, which will serve as a helpful conversion tool. You may also want to utilize this program as a testing machine, having a person figure out conversion problems and then testing his or her answers against the computer's.

Important Variables in Length Converter

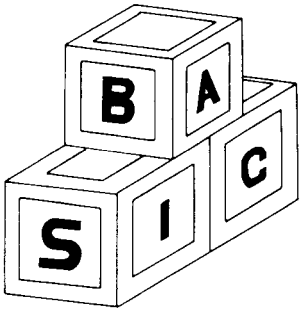
INP$	Input to see if user wants to continue
UNIT1	The unit of measure that the user wants to convert
AMOUNT1	The amount of this unit
UNIT2	The unit that to which the user wants to change
AMOUNT2	The amount of the above

Important Lines in Length Converter

50-370	Initialization/Prints Menu
380	Goes to selected unit
400-1070	Converts to selected unit
4000-4060	Gets necessary input
5000	Gives user the answer
5010-5030	Waits until the user wants to continue

This completes our exploration of how the computer can work with math. After using these programs, you will discover that your ATARI is an excellent tool for mathematics. You may write some programs yourself covering other areas of math, such as calculus, graphing, and geometry—accompanied by graphics to show the various geometric shapes and their formulae.

Chapter 5



Graphics Programs for the ATARI

The ATARI home computers can produce high-quality color graphics with only a few simple commands. It is certainly worth our while to take a look into the power of the graphics of the ATARI and explore some of its uses.

TERMS YOU NEED TO KNOW

Before we begin examining the BASIC programs in this chapter that work with the graphics of the ATARI, there are a few terms you need to know so you'll understand what I am talking about when I mention such things as *hue, luminance,* and *high resolution.*

Color. There are a total of 256 different colors you can utilize on the ATARI home computers, although some graphics modes don't support this many colors. In fact, in the very intricate and detailed graphics modes, as few as two colors may be available.

Graphics. These are the pictures (as opposed to the words and numbers) you see on the screen as in Fig. 5-1. Bar graphs, line graphs, spaceships in arcade games, and any other types of visual images you see can be called graphics.

Low Resolution. Graphics which are rough, not well-defined, and not accurately representative of what they are supposed to be (for instance, graphics showing a ship which doesn't exactly look like a ship) are *low-resolution graphics.* These don't take up very much memory and are quite simple to program, but they are

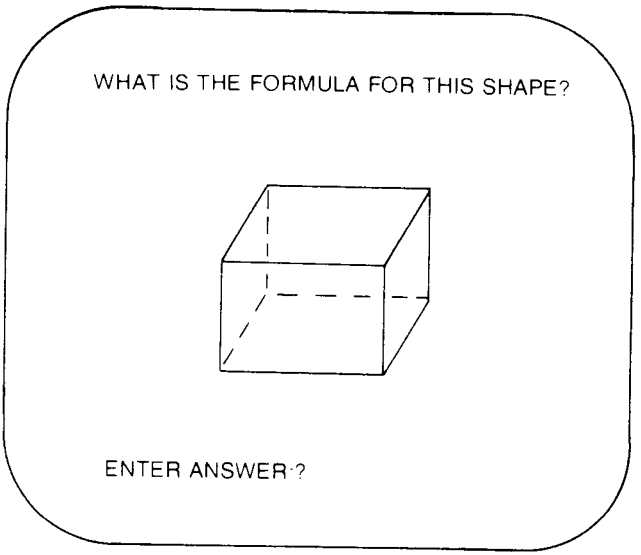


Fig. 5-1. A screen showing graphics.

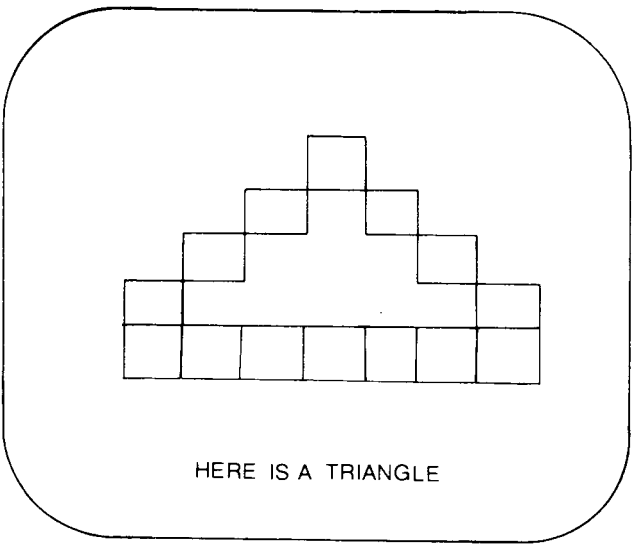


Fig. 5-2. Low-resolution graphics.

low in quality and aren't appropriate for things such as arcade games or line graphics, which generally require high resolution. For things such as bar graphs or simple charts, however, low resolution usually will suffice. Figure 5-2 is an example of low resolution.

High Resolution. This is the opposite of low resolution, since these graphics are sharp, very well-defined, and take up a significant amount of memory, time, and effort to complete and draw. Figure 5-3 shows high-resolution graphics.

Hue. This is a variation of a color, Its gradation along the spectrum of visible light from red to violet. With a change in hue, a color's appearance will also change.

Luminance. This is the "brightness" of a color. If you increase the luminance, the brightness increases.

Now that we have learned the definitions of these terms, we can take a look at the four different programs in this chapter.

COLORBAR

Using this first program is an excellent way to learn the significance of hue and luminance in affecting the way a color appears on the screen. To use this program, simply press the H

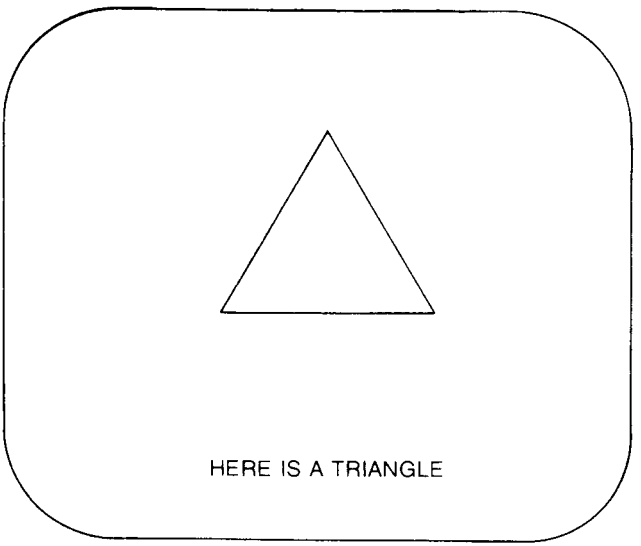


Fig. 5-3. High-resolution graphics.

key to increase the value of the hue, SHIFT-H to decrease the value of the hue, L to increase the luminance, and SHIFT-L to decrease the luminance.

The first part of the program puts the ATARI into graphics mode, makes certain that the values of hue and luminance are zero, and shows the values of both hue and luminance:

```
50   GRAPHICS 1

60   HUE=0

70   LUM=0

100  POSITION 1,1

110  PRINT "HUE IS ";HUE

115  PRINT "LUMINANCE IS ";LUM
```

At this point, color register four is set to the values of HUE and LUM; the ATARI begins monitoring the keyboard in case you press anything. If you press H, SHIFT-H, L, or SHIFT-L, the computer will go to the appropriate line number. If not, it will go back to line 130 to go through the process again.

```
120  SETCOLOR 4,HUE,LUM

130  A=PEEK(764)

140  IF A=57 THEN 200

150  IF A=0 THEN 300

160  IF A=121 THEN 400

170  IF A=64 THEN 500

180  GOTO 30
```

The different routines for changing hue and luminance are all basically the same. When the computer has finished making the changes in the variable specified, it goes to line 600 to "clear out" the value in variable A. This keeps it from "thinking" that the key you just pressed is being pressed again.

```
200  IF HUE=15 THEN 600

210  HUE=HUE+1

220  GOTO 600
```

```
300  IF LUM=14 THEN 600
310  LUM=LUM+2
320  GOTO 600
400  IF HUE=0 THEN 600
410  HUE=HUE-1
420  GOTO 600
500  IF LUM=0 THEN 600
510  LUM=LUM-2
520  GOTO 600
600  A=255
610  POKE 764,255
615  PRINT
620  GOTO 100
```

As you can see at line 620, the program keeps looping back to a point near the beginning so you can change the hue and luminance repeatedly. This program is helpful for finding the particular shades of color you need for your programs and also for learning about the significance of hue and luminance. You also might want to try changing the first value in the SETCOLOR statement (the register number, which is shown here as 4) to a different number to see the results.

Important Variables in Colorbar

HUE	Hue Value of SETCOLOR command
LUM	Luminance value
A	Value of key pressed

Important Line Numbers in Colorbar

110-115	Tells user the color values
120	Sets the color
130-180	Gets keyboard value
200-220	Adds 1 to hue value
300-320	Adds 2 to luminance value
400-420	Subtracts 1 from hue value
500-520	Subtracts 2 from luminance value
600-620	Resets keyboard value

SHAPES

Here is a simple program to teach you how to draw certain shapes, like three-dimensional cubes and two-dimensional triangles, on the video screen. The program will display a menu of four different shapes, plus an option to end the program. If you select a shape, it will be shown with the vertices used to draw it. If you want to end the program, simply select Option 5 and the program will stop. Figure 5-4 shows a sample rectangle.

At the beginning of the program, room is made for the C$ variable, the screen is cleared, and the menu, is displayed:

```
5    DIM C$(1)
10   GRAPHICS 0
20   POSITION 7,7
30   PRINT "SAMPLE SHAPES LIBRARY"
40   POSITION 9,10
50   PRINT "1. RECTANGLE"
60   POSITION 9,11
70   PRINT "2. TRIANGLE"
```

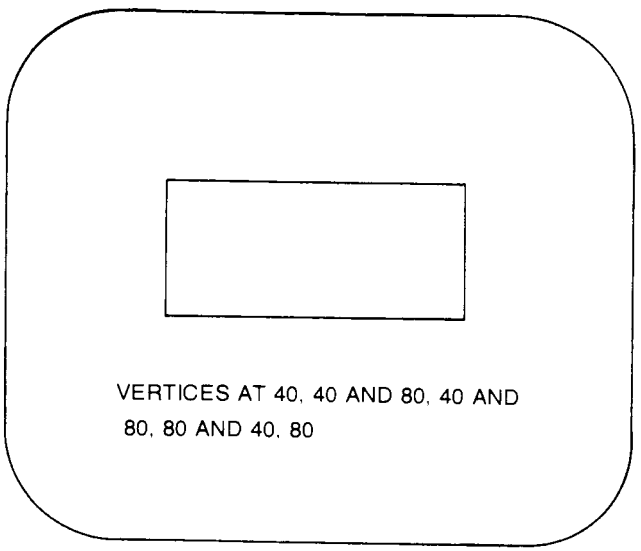


Fig. 5-4. Sample screen from Shapes.

```
80   POSITION 9,12
90   PRINT "3. BOX"
100  POSITION 9,13
110  PRINT "4. ABSTRACT"
120  POSITION 9,14
130  PRINT "5. END"
```

The program allows you to input your choice; it then acts on the value of your choice by going into high-resolution graphics mode, establishing a color for graphics, and going to the appropriate line number.

```
140  INPUT CHOICE
145  GRAPHICS 14
147  COLOR 3
150  ON CHOICE GOTO 200,300,400,500,600
```

The shapes below are rectangle, triangle, box, and abstract, respectively. As you can see, the PLOT and DRAWTO statements are used to construct the shapes, while the PRINT statement displays the vertices used to draw the shape.

```
200  PLOT 40,40
210  DRAWTO 80,40:DRAWTO 80,80
220  DRAWTO 40,80:DRAWTO 40,40
230  PRINT "VERTICES AT 40,40 AND 80,40"
240  PRINT "AND 80,80 AND 40,80."
250  GOTO 700
300  PLOT 80,20
310  DRAWTO 100,80:DRAWTO 60,80
320  DRAWTO 80,20
330  PRINT "VERTICES AT 80,20 AND 100,80"
340  PRINT "AND 60,80."
350  GOTO 700
400  PLOT 40,40
```

```
410  DRAWTO 80,40:DRAWTO 80,80
420  DRAWTO 40,80:DRAWTO 40,40
430  DRAWTO 60,20:DRAWTO 100,20
440  DRAWTO 100,60:DRAWTO 80,80
445  PLOT 80,40
450  DRAWTO 100,20:PLOT 40,80
460  DRAWTO 60,60:DRAWTO 100,60
470  PLOT 60,60:DRAWTO 60,20
480  PRINT "VERTICES AT 40,40 AND 80,40 AND 80,80"
490  PRINT "AND 40,80 AND 60,20 AND 100,20 AND "
492  PRINT "100,60 AND 60,60"
495  GOTO 700
500  PLOT 20,20
510  FOR LOOP1=1 TO 30
520  X=INT(160*RND(1)+1)
530  Y=INT(160*RND(1)+1)
540  DRAWTO X,Y
550  NEXT LOOP1
560  PRINT "30 RANDOM VERTICES"
570  GOTO 700
```

Option 5, "End Program," simply clears the screen and ends the program.

```
600  GRAPHICS 0
610  END
```

On the other hand, if you have selected a shape, you can draw another one by pressing C and then <RETURN> in answer to the question below:

```
700  PRINT"INPUT C TO CONTINUE.";
710  INPUT C$
720  GOTO 10
```

When you are drawing different shapes, you certainly don't have to use the vertices shown here, but you should examine the relationships of the vertices so you can draw similar shapes in your own programs.

Important Variables in Shapes

CHOICE	Input user's choice of a shape
LOOP1	Loop used to make a random, abstract shape
X,Y	Coordinates for abstract pattern
C$	Input used to play again or not

Important Line Numbers in Shapes

5-150	Initialization to get input from user and go to chosen shape
200-250	Draws rectangle
300-350	Draws triangle
400-495	Draws a box
500-570	Draws an abstract figure
600-610	Ends program
700-720	Asks for input to continue or not

DRAWER

For sketching graphics on the screen, as in Fig. 5-5, you can use this short program that allows you to use the following keys for the following functions:

KEY	FUNCTION
Y	Move Drawer up
B	Move Drawer down
G	Move Drawer left
H	Move Drawer right
Q	Turn color to cyan
W	Turn color to orange
E	Turn color to blue
R	Go to "non-drawing" mode

The program begins by going into full-screen high-resolution graphics mode and assigning COLOR 2 to the graphics, while the X and Y values are set to zero:

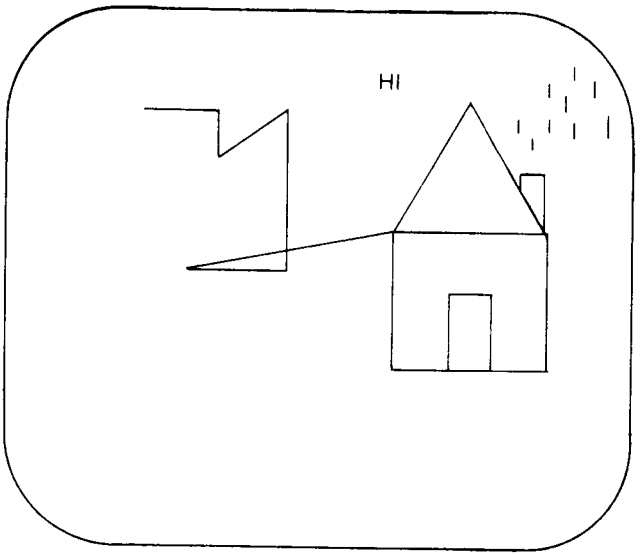


Fig. 5-5. A drawing program example.

```
10   GRAPHICS 7+16
20   COLOR 2
30   X=0:Y=0
```

Then, the point in the upper left-hand corner of the screen is drawn, and the keyboard byte (764) is "cleared" by poking a value of 255 into it:

```
40   PLOT 0,0
100  POKE 764,255
```

At this point, the computer begins monitoring byte 764. Any activity in that byte will be interpreted by the computer and acted upon by the ATARI:

```
110  K=PEEK(764):IF K=255 THEN GOTO 110
120  IF K=47 THEN COL=2:GOTO 400
```

```
130  IF K=46 THEN COL=1:GOTO 400
140  IF K=42 THEN COL=3:GOTO 400
145  IF K=40 THEN COL=0:GOTO 400
150  IF K=43 THEN Y=Y-1:GOTO 200
160  IF K=61 THEN X=X-1:GOTO 200
170  IF K=57 THEN X=X+1:GOTO 200
180  IF K=21 THEN Y=Y+1:GOTO 200
190  GOTO 100
```

If the user happens to draw beyond the boundaries of the screen, the ATARI automatically corrects the mistake:

```
200  IF Y<0 THEN Y=0
210  IF Y>95 THEN Y=95
220  IF X<0 THEN X=0
230  IF X>159 THEN X=159
```

In addition, if the current color of the cursor is 0 (blank), then a special routine at line 500 is accessed to blink the drawer temporarily. Otherwise the graphics are drawn on the screen and the computer goes back to the "keyboard detect" sequence.

```
235  IF COL=0 THEN GOSUB 500
240  DRAWTO X,Y
250  GOTO 100
400  PLOT X,Y
420  COLOR COL
430  GOTO 100
500  COLOR 2
510  PLOT X,Y
520  FOR DELAY=1 TO 50:NEXT DELAY
530  COLOR 0
540  PLOT X,Y
550  RETURN
```

When you want to move around the screen without drawing, go into the "non-draw" mode. Otherwise, select a color of your choice and draw with that. This program can be helpful when you are developing graphics or just experimenting with the potential of the ATARI home computer.

Important Variables in Drawer

X,Y	Plot values
K	Keyboard value
COL	Color register value
DELAY	Used for delay loop

Important Lines in Drawer

10-40	Initialization
100-190	Gets keyboard input
200-230	Sees if drawer is going off the edge
235	If drawing in erase mode, computer makes a point blink to show where the drawer is.
240	Does actual drawing
400-430	Changes drawer to desired color
500-550	When in erase mode, makes a point blink to show position

GRAPH

This program has a practical purpose in that you can use it for inputting data and having that data displayed in line graph form similar to Fig. 5-6. To use this program, simply type in the number of data items you will be entering (up to 10), then enter each value, which may range from 0 to 159. Once you have entered all of your numbers, the computer will plot it on the graph; you then have the option to draw another one.

First of all, the program sets aside room for its variables, clears the screen, and asks how many items there are to be entered:

```
5    DIM ITEM(10)

7    DIM Y$(1)

10   GRAPHICS 0

20   PRINT "HOW MANY ITEMS? (UP TO 10)"

30   INPUT ITEMS
```

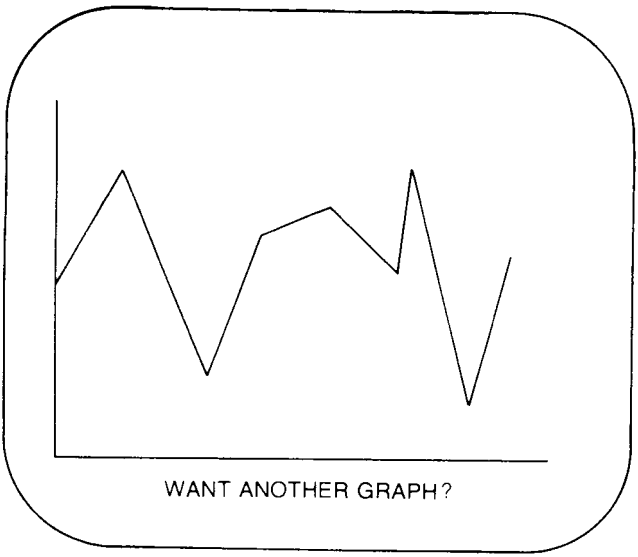


Fig. 5-6. A line graph.

Then a loop begins so you can enter each of the items:

```
40   FOR LOOP1=1 TO ITEMS
50   PRINT "ENTER AMOUNT OF ITEM ";LOOP1
60   PRINT "(UP TO 159)"
70   INPUT I
80   ITEM(LOOP1)=I
90   NEXT LOOP1
```

Once you have put in all of your items, the ATARI computes how much space on the X-axis should be allocated for each item of data. It then goes into high-resolution graphics mode and draws the two axes:

```
100  ADDX=INT(159/ITEMS)
110  GRAPHICS 14
120  COLOR 1
125  PLOT 0,0
```

```
130  DRAWTO 0,159

140  DRAWTO 159,159
```

For graphing, the computer simply plots the first point and then goes to each successive data item you gave it using the DRAWTO statement. When it has finished you have your line graph. The computer asks you if you would like another graph, the answer to which question causes it to respond accordingly.

```
150  X=0

160  PLOT X,159-ITEM(1)

170  FOR LOOP2=2 TO ITEMS

180  DRAWTO X+ADDX,159-ITEM(LOOP2)

190  X=X+ADDX

200  NEXT LOOP2

210  PRINT "WANT ANOTHER GRAPH":INPUT Y$

220  IF Y$="Y" THEN RUN

230  GRAPHICS 0
```

You might also try modifying this program to support bar graphs or line graphs that are "filled in" below the graph lines, so that a solid exists where there was space before.

Important Variables in Graph

ITEM (X)	Data for items on X-axis
ITEMS	Input for number of items on X-axis
LOOP1	Used to ask for input amount for above
I	Input for entering data
ADDX	Used when dividing up the X-axis so there will be enough room for all pieces of data
LOOP2	Used to graph each amount on the grid
X	Number of positions on X-axis available
Y$	Sees if you want to draw another graph or not

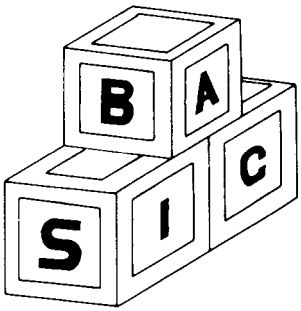
Important Line Numbers in Graph

20-90	Gets information for the graph
100	Gets amount to add to the value for the X-axis each time a new point in drawn

120-140	Draws both axes
150-160	Graphs first item
170-200	Graphs other items
210-230	Checks to see if you want to draw another graph

These programs have only begun to touch upon the ATARI's graphics capabilities, but they do give you an idea of the uses for this particular aspect of the computer. You might also want to try experimenting with the ATARI graphics commands, working out program ideas that you could use for learning geometry, helping in business, or creating educational software for younger people. Whatever your purpose, the ATARI graphics capabilities should serve you well—as long as you use the available memory efficiently.

Chapter 6



Games

The ATARI home computers are probably best known for their ability to support high-quality games. These games, most of them "arcade-type," have provided hours of entertainment for home computer owners—especially considering how much better computer games are over home video game systems.

In this chapter, we will be taking a look at three game programs. Because the BASIC language isn't as fast and efficient as the machine language in which most arcade games are written, these games are not "arcade-type." Nevertheless, they are still entertaining games which will teach you more about how BASIC may be used.

GUESS MY NUMBER

The first game, a very basic one at that, is called "Guess My Number," and it involves trying to guess a random number the computer has generated and stored in its memory. The number can range from 1 to 100. Once you have guessed the number, the computer will tell you how many guesses it took your deductive reasoning to find the number.

The program itself is extremely simple, since it uses primarily the RND and the INPUT statements. The first line creates a string, YN$, that can hold one character:

```
1     DIM YN$(1)
```

Then the screen is cleared and the variable which records how many tries have been made is set to 1:

```
2    GRAPHICS 0
5    TIMES=1
```

The random number is then generated by the RND statement, and the computer asks the user to guess the number.

```
10   NUMBER=INT(100*RND(1)+1)
20   PRINT"WHAT IS MY NUMBER?
30   INPUT GUESS
```

Once the guess has been input, the computer checks whether it is correct, too low, or too high. If the guess is greater than 100, it returns to the question at line 20. Otherwise, the computer will check the condition of the number and go to the appropriate line number:

```
40   IF GUESS>100 THEN 20
50   IF GUESS>NUMBER THEN 80
60   IF GUESS<NUMBER THAN 110
70   IF GUESS=NUMBER THEN 140
```

Once it reaches the line number, the computer takes appropriate action for whatever the guess might be. If the guess is too high, the computer goes to this routine:

```
80   PRINT "MY NUMBER IS LOWER"
90   TIMES=TIMES+1
100  GOTO 20
```

If the guess is too low:

```
110  PRINT "MY NUMBER IS HIGHER"
120  TIMES=TIMES+1
130  GOTO 20
```

Finally, if the guess is correct, the computer tells you how

many guesses it took you to find the correct number. It then asks whether you would like to play again or not, taking appropriate action on your response:

```
140  PRINT:PRINT
150  PRINT"YOU GOT MY NUMBER RIGHT IN ";TIMES;" GUESSES."
160  PRINT"DO YOU WANT TO PLAY AGAIN?"
170  INPUT YN$
180  IF YN$="Y" THEN 2
```

This completes the program. When you are playing it, try to see how few guesses it takes you to guess the computer's number. A sample screen is shown in Fig. 6-1.

Important Variables in Guess My Number

YN$	Input to see if player wants to play again or not
TIMES	Used to count number of guesses the player needed to guess the correct number

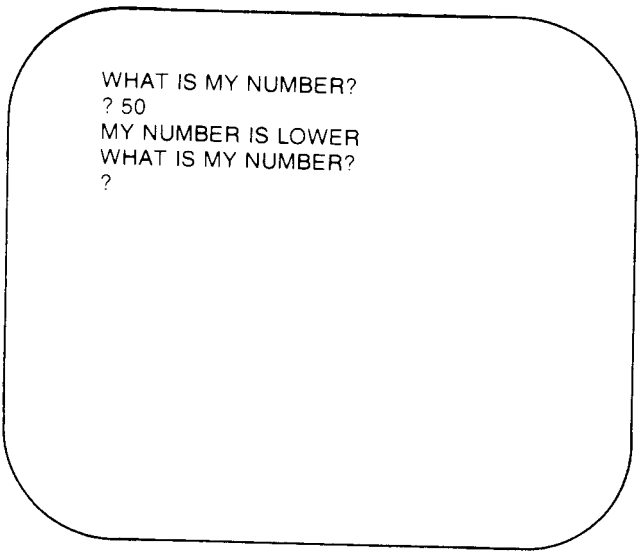


Fig. 6-1. Guess My Number.

NUMBER	A random number between 1 and 100
GUESS	The player's Guess

Important Line Numbers in Guess My Number

10	Gets random number
20-30	Gets player's guess
40-70	Checks to see what player guesses
80-130	Tells player whether number is higher or lower than his guess

SIMON SAYS

This program is a memory game, since you must remember a sequence of colors and sounds the computer gives you. You select the difficulty level (how long the colors and sounds are shown and played) along, with the number of colors and sounds you must remember. The computer shows and plays the colors and sounds, and then you tell it what you remember. Afterwards you may play again, perhaps to try a more difficult level, or stop the game.

To set up the game, I programmed the computer to set up the variables RAN and RESPONSE$ and clear the screen:

```
10   DIM RAN(25)

20   DIM RESPONSE$(1)

50   GRAPHICS 0
```

The computer then asks how many screens were wanted. The more screens there were, the harder it will be to remember all of them.

```
60   POSITION 4,9

70   PRINT"HOW MANY SCREENS DO YOU WISH TO TRY?"

72   POSITION 4,13

75   INPUT TIMES
```

The computer is then programmed to ask what difficulty level is desired. The difficulty level determines what period of time the color and sound will be shown and played. The higher

the number the longer the period, and thus the easier it is to remember the sequence.

```
80   POSITION 4,15
85   PRINT"ENTER DIFFICULTY LEVEL (1-3)"
90   POSITION 4,16
100  PRINT"LEVEL 1 IS THE MOST DIFFICULT"
110  INPUT LEVEL
120  DELAY=LEVEL*100
```

The computer then clears the screen and begins the loop to select the random colors:

```
130  GRAPHICS 0
140  FOR LOOP1=1 TO TIMES
```

Once the loop is running, random numbers (between 1 and 4) are selected, the screen is cleared, and the computer accesses the appropriate combination of color and sound in accordance with the randomly chosen number.

```
200  RAN(LOOP1)=INT(RND(1)*4)+1
210  GRAPHICS 0
220  ON RAN(LOOP1) GOTO 300,400,500,600
```

The routines for the four different colors and sounds are as follows:

```
300  SETCOLOR 2,2,8
310  SOUND 0,100,14,15
320  GOTO 700
400  SETCOLOR 2,12,8
410  SOUND 0,50,8,15
420  GOTO 700
500  SETCOLOR 2,8,8
```

```
510  SOUND 0,50,14,15
520  GOTO 700
600  SETCOLOR 2,4,5
610  SOUND 0,250,14,15
```

The routine at line 700 delays for a short while so the color and sound will remain on. Then the computer clears the sound currently being played and checks to see if it should play another sound and show another color. If so, it continues the LOOP1. If not, it goes to line 800.

```
700  FOR LOOP2=1 TO DELAY:NEXT LOOP2
710  SOUND 0,0,0,0
720  IF LOOP1=TIMES THEN 800
730  NEXT LOOP1
```

At line 800, the computer begins the loop, which asks you which color was on the screen. It will continue asking you this until you have responded with the correct answer to all of the colors. Once it has finished this loop, it will move on to the rest of the program.

```
800  FOR LOOP3=1 TO TIMES
810  GRAPHICS 0
820  POSITION 6,6
830  PRINT"1. ORANGE"
835  POSITION 6,7
840  PRINT"2. GREEN"
845  POSITION 6,8
850  PRINT"3. VIOLET"
855  POSITION 6,9
860  PRINT"4. RED"
870  POSITION 4,11
880  PRINT"WHICH WAS COLOR SCREEN NUMBER ";LOOP3
890  INPUT ANSWER
```

```
900  IF ANSWER<>RAN(LOOP3) THEN 1000
910  NEXT LOOP3
```

The remainder of the program either congratulates the player for answering all of the questions correctly or tells the player he input a wrong answer. After a win or lose has been called, the computer asks if the player would like to play the game again, so that it may take appropriate action.

```
915  PRINT:PRINT"GOOD JOB!"
920  PRINT:PRINT
930  PRINT"DO YOU WANT TO PLAY AGAIN?"
940  INPUT RESPONSE$
950  IF RESPONSE$="Y" THEN RUN
960  END
970  POSITION 6,14
1010 PRINT"WRONG...YOU MISSED THAT COLOR."
1020 GOTO 920
```

This game is good for working for memory and you should try to see just how many colors and sounds you can retain in your mind. You also might want to try increasing the difficulty level for added challenge in the game.

Important Variables in Simon Says

RAN(X)	Random number (1 through 4) corresponding to a color
RESPONSE$	Your response to whether you want to play again or not
TIMES	Input for number of screens to try
LEVEL	Input for level of difficulty
LOOP1	Loop to display the random colors
DELAY	Used to delay loop in accordance with the choice picked when "Enter difficulty level (1-3)" is asked
LOOP2	Delay loop
LOOP3	Asks for a certain number of answers, equivalent to the value of the variable TIMES

ANSWER	Input for the color the player thinks is correct

Important Line Numbers in Simon Says

70-100	Gets input from player
120	Sets length of delay used when showing each color
140-730	Displays a certain number of colors equivalent to the variables TIMES
200	Gets a random color (SETCOLOR value ranging from 1 to 4)
220	Goes to randomly selected line number
700	Delay to display color on the screen for a certain length of time
710	Turns off sound
800-910	Asks play what each color was
915	Tells player he got everything right
920-960	Asks "Do you want to play again?"
1010	Tells player that he did not guess a color correctly

BLACKJACK

Blackjack or "21" is a popular card game in which you try to beat the "dealer" (the ATARI) by getting a score as close to the number of 21 as possible, without going over. You can also get a jack and an ace, which is an automatic win called "blackjack" (Fig. 6-2). Of course, the dealer could get blackjack or get closer to 21 than you after you decide not to "take another hit" (get another card). Simply try to get as close to 21 as you can, but avoid a "bust," which occurs when you go over 21. Keep in mind that the values for the cards are the same as their face value; the jack, king, and queen have a value of 10, while the ace can have values of 11 or 1.

"Blackjack" begins by setting up the variables for the program and clearing the screen:

```
10   DIM RAN(52):DIM DEALT(52)

15   DIM CARD(52):DIM A$(2)

20   GRAPHICS 0

30   MONEY=100
```

The computer then "shuffles" the cards by assigning each card a unique value. This routine takes a while, since it must read data from lines 2000 through 2030 (also shown below) and make certain that the card it selects for variable is unique—so you won't wind up with a deck of 15 queens.

```
40   PRINT"SHUFFLING"

45   NM=0

50   FOR LOOP3=1 TO 52

60   DEALT(LOOP3)=0

70   NEXT LOOP3

100  FOR LOOP1=1 TO 52

110  RAN(LOOP1)=INT(52*RND(1)+1)

120  FOR LOOP2=1 TO 51

140  IF RAN(LOOP1)=DEALT(LOOP2) THEN 110

150  NEXT LOOP2
```

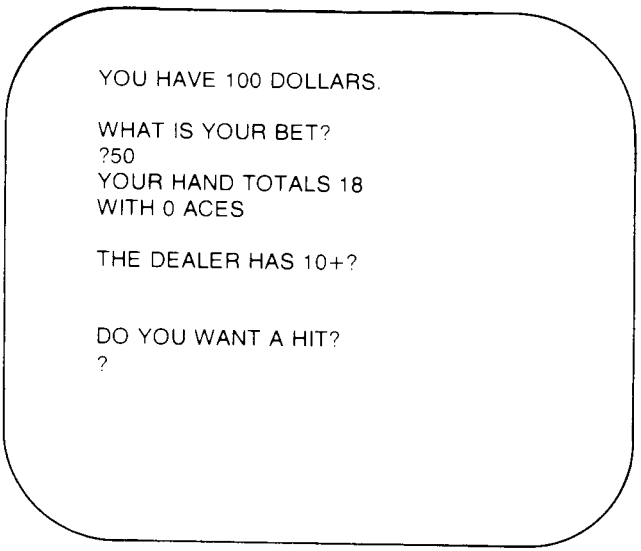


Fig. 6-2. Blackjack.

```
160  DEALT(LOOP1)=RAN(LOOP1)
170  FOR LOOP4=1 TO RAN(LOOP1)
180  READ CARD
190  NEXT LOOP4
200  CARD(LOOP1)=CARD
210  RESTORE
220  NEXT LOOP1
2000 DATA 11,10,10,10,10,9,8,7,6,5,4,3,2
2010 DATA 11,10,10,10,10,9,8,7,6,5,4,3,2
2020 DATA 11,10,10,10,10,9,8,7,6,5,4,3,2
2030 DATA 11,10,10,10,10,9,8,7,6,5,4,3,2
```

The computer clears the screen, determines that the number of aces the player and the dealer have is zero, and then checks to see if there are enough cards with which to play. If there are not, the computer goes to a reshuffling routine at line 1700.

```
300  GRAPHICS 0
303  PA=0:DA=0
305  IF NM>37 THEN 1700
```

In order for a person to know how much money is available for betting, the computer shows the amount of money the player has, then accepts a bet from the player:

```
310  PRINT"YOU HAVE ";MONEY;" DOLLARS."
320  PRINT
330  PRINT"WHAT IS YOUR BET?"
340  INPUT BET
350  IF BET>MONEY THEN 330
360  MONEY=MONEY-BET
```

The computer gives the player his hand of cards, totals their value, and then tells the player the value of his cards and how many aces he has.

```
370  NM=NM+1
380  PHAND=CARD(NM)+CARD(NM+1)
390  IF CARD(NM)=11 THEN PA=PA+1
400  IF CARD(NM+1)=11 THEN PA=PA+1
405  NM=NM+2
410  IF PHAND=21 THEN 1000
420  IF PHAND>21 THEN GOSUB 1100
430  PRINT"YOUR HAND TOTALS ";PHAND
440  PRINT"WITH ";PA;" ACES."
```

The computer then does the same for the dealer:

```
500  DHAND=CARD(NM)+CARD(NM+1)
510  IF CARD(NM)=11 THEN DA=DA+1
520  IF CARD(NM+1)=11 THEN DA=DA+1
525  NM+2
530  IF DHAND=21 THEN 1200
540  IF DHAND>21 THEN GOSUB 1300
550  PRINT:PRINT
560  PRINT"THE DEALER HAS:"
570  PRINT CARD(NM);" + ?"
600  PRINT:PRINT
```

The player is then asked if he wants a hit. If he does, he is given another card and the value of the hand is checked again. If he does not want another card (i.e. "stand"), the computer will go to the routine at line 750. Figure 6-2 outlines the dialogue up to this point.

```
610  PRINT"DO YOU WANT A HIT?"
620  INPUT A$
630  IF A$<>"Y" THEN 750
640  PHAND=PHAND+CARD(NM)
650  NM=NM+1
```

```
660  IF CARD(NM-1)=11 THEN PA=PA+1
670  IF PHAND>21 THEN GOSUB 1100
680  PRINT"YOUR HAND TOTALS ";PHAND
690  PRINT"WITH ";PA;" ACES."
700  GOTO 600
```

If the player decides to stand, the computer will show the dealer's hand and will compare the value of the dealer's hand with the value of the player's hand, taking appropriate action for whatever the condition might be. For example, if the dealer requires a hit, the computer will give him one. If the dealer has a good hand, the computer will go to the routine at line 1400, showing that the dealer has won.

```
750  PRINT:PRINT
760  PRINT"THE DEALER'S HAND TOTALS ";DHAND
770  PRINT"HE HAS ";DA;" ACES."
780  IF PHAND>DHAND THEN 900
790  IF DHAND=PHAND THEN 900
800  IF DHAND>PHAND THEN 1400
900  IF DHAND>16 THEN 1500
```

If the computer decides to go to line 910, the dealer receives a hit, checks his hand, then goes back to the previous routine to see if he requires another hit.

```
910  DHAND=DHAND+CARD(NM)
920  IF CARD(NM)=11 THEN DA=DA+1
930  NM=NM+1
940  IF DHAND>21 THEN GOSUB 1300
950  GOTO 760
```

In the unlikely event that the player is dealt a blackjack, the computer informs that player that he has indeed acquired a 21-point hand, allowing him to win triple his bet money—and the game.

```
1000 PRINT:PRINT"BLACKJACK"
1010 BET=BET*3
1020 MONEY=MONEY+BET
1030 GOTO 1600
```

If the player's cards have a value over 21, the aces are reduced from 11 to 1 by the computer, as shown below:

```
1100 IF PA<1 THEN 1400
1110 PA=PA-1
1120 PHAND=PHAND-10
1130 RETURN
```

Of course, the dealer can also get a blackjack. The routine below allows for that possibility, and also includes the equivalent of the "ace routine" shown above.

```
1200 PRINT:PRINT "THE DEALER HAS 21."
1205 IF MONEY=0 THEN 1900
1210 GOTO 1600
1300 IF DA<1 THEN 1500
1310 DA=DA-1
1320 DHAND=DHAND-10
1330 RETURN
```

Finally, here are the routines for winning and losing the game, along with the lines which ask the player if he would like to play the game again. The last few lines are for the purpose of "reshuffling" the deck, in case the cards remaining are too few in number:

```
1400 PRINT:PRINT"YOUR HAND TOTALS ";PHAND
1410 PRINT"THE DEALER HAS ";DHAND
1415 PRINT"YOU LOSE."
1417 IF MONEY=0 THEN 1900
1420 GOTO 1600
```

```
1500 PRINT:PRINT"YOUR HAND TOTALS ";PHAND
1510 PRINT"THE DEALER HAS ";DHAND
1520 PRINT"YOU WIN!"
1530 MONEY=MONEY+BET*2
1600 PRINT:PRINT"DO YOU WANT TO PLAY AGAIN?"
1610 INPUT A$
1620 IF A$="Y" THEN 300
1630 END
1700 PRINT:PRINT"SORRY, I MUST RE-SHUFFLE."
1705 MONEY=MONEY+BET
1710 GOTO 45
```

Important Variables in Blackjack

RAN(X)	Random number to read off a random amount of data
DEALT(X)	Used to make sure that the same RAN(X) has not been used twice
CARD(NM)	Value of Xth card in shuffled deck
A$	Input to see if player wants to play again
MONEY	Amount that player has in dollars
LOOP3	Sets DEALT(X) values to zero
LOOP1	Loop for card shuffling routine
CARD	Card values read from data
LOOP2	Makes sure that same RAN(X) is not used twice
LOOP4	Used to read data
PA	Number of aces the player has
DA	Number of aces the dealer has
BET	Amount that player bets
PHAND	Total value of player's hand
DHAND	Total value of dealer's hand

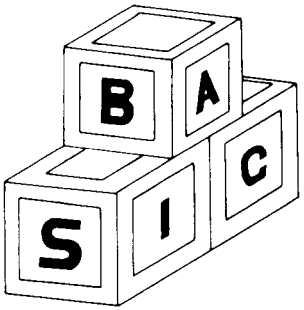
Important Line Numbers in Blackjack

10-45	Initialization
50-70	Clearing DEALT(X)
100-200	Entire shuffling routine
110	Get random value
120-150	Makes sure value has not been used before

160	Stores random value for use in 120-150
170-190	READs random number of card values
200	Gets value of card
305	Goes to line 1700 if there are not enough cards with which to play
310-340	Gets player's bet
350	Makes sure the player doesn't bet more than he has
370-440	Gives player his hand, counts number of aces, totals its value, see if it's over 21 or 21 even
500-570	Does same for the dealer
600-630	Sees if player wants a hit
640-690	Gives player a hit, checks value of hand
700	Goes to ask if player wants a hit again
750-900	Sees if dealer needs a hit, or if he has 17 or over
910-950	Gives dealer a hit, checks his hand, goes back to see if he needs another hit
1000-1030	Tells player he got blackjack, asks if he wants to play again after adding to his money
1100-1130	This routine checks to see if the player has any aces since he has over 21. Aces are therefore changed from 11 to 1
1200-1210	Informs the player that the dealer dealt himself 21, sees if player lost all of his money or not
1300-1330	Ace routine for the dealer
1400-1420	Informs the player that he has lost
1500-1530	Informs the player that he has won and adds his bet to the money he already has
1600-1620	Sees if player wants to play again.

Now that we have taken a look at just about every type of program we can make on the ATARI home computer, and are knowledgeable in the BASIC language and how it can be utilized, we can complete our learning and programming experience. Let's see how you can make your own ideas for your own programs and then develop them into actual, working pieces of software.

Chapter 7



Creating Your Own Programs

Up to now you have been entering and using the programs in this book, discovering the various keywords used in creating programs, and finding out new uses for your computer. Along the way, you have probably devised some program ideas of your own, and you might have modified the programs in this book for your own purposes or experimentation. If you have, this is an excellent first step to becoming "your own programmer." If you haven't, you might want to try to develop your programming skills now by reviewing some of the programs you have typed in and making some changes, no matter how insignificant.

The main point is that helping you acquire the ability to create your own programs is one of the main purposes of this book. You should learn at least some of the BASIC keywords of the ATARI and know the proper way to put the words together to make a usable program. More importantly, you should learn the technique behind creating a program. This last aspect has not yet been covered, so we will explore it in this final chapter.

THE USEFUL COMPUTER

Because one of the most important aspects of creating a program is coming up with an original program idea, we should first look at the different types of programs so you can find an idea which might fit into one of these types. Reading about these

various kinds of programs will probably make it easier for you to develop an idea for a program.

The first (and probably most obvious) type of program for the 600XL or 800XL computer system is *entertainment software.* Their high-resolution graphics, excellent sounds, and relatively large memory make them excellent machines on which to play games. There have already been a wide variety of computer games developed for the ATARI home computers, and here are some types that have already come into existence for the 600XL and 800XL.

Games of Chance. These might be called "gambling games," since they involve themes that would usually be found in Las Vegas. "One-Armed Bandit" (a slot machine), Blackjack, and Keno are just a few. Computers are excellent for playing card games and games of chance, especially the 600XL or 800XL with color graphics (useful for displaying cards, slot machines, or dice), sounds (for indicating winning or losing), and large memory (for storing a great deal of information).

Adventures. An adventure is a game in which you play the part of a person in a strange situation (such as on a deserted island, on a faraway planet, or in a haunted house) who must use skill and luck to retrieve treasures, stay alive, or (usually) both. Adventures are enjoyable because they involve strategy and wit, and they can put you in a fantasy world which only your computer and your imagination can produce.

Arcade Games. These are certainly the most popular of all kinds of games, since they combine exciting graphics, sounds, and fast action to produce a game requiring fast reflexes—but little intelligence. One look at any magazine which includes the ATARI home computers in its subject matter will show you that there are many arcade games currently available for the ATARI.

Board Games. Some games (like "Life" from Milton Bradley and "Monopoly" from Parker Brothers) commonly played on boards can be adapted for use on the computer. In fact, some games exist for other computers which let the computer play against you in a board game.

Logic Games. Games which require a lot of logical thinking (like "Mastermind," which requires you to pick out a pattern of four colored pegs) are popular on computers, since programmers and other people who enjoy computers are usually logical folks who enjoy using their mind to solve problems.

This covers most of the different kinds of games which are

available for the 600XL and 800XL computer systems. Another type of software which give the 600XL or 800XL a useful application besides entertainment is *educational software.*

Learning from a computer is an extremely exciting possibility. Children (and adults) can discover foreign languages, advanced mathematics, and excellent spelling habits by utilizing a computer able to show them rules, test them, quiz them with repetitive questions and have an infinite amount of patience and encouragement. Human teachers certainly can do a lot better teaching some subjects than computers can, but for repetitive teaching or education which doesn't require a lot of creativity or customized training by the educator, computers are excellent for instructing people in a wide variety of subjects.

There are several types of programs available today which can help many young people and adults in learning about math, science, spelling, or practically any other subject.

Computer Literacy. One of the best things that computers like the 600XL or 800XL can teach is, naturally, how to work with computers. Programs can teach how to use the BASIC language, how to use the keyboard, what the different parts of a computer are, how to use a word processor, and what programming is involved in producing graphics and sounds. As I have mentioned before, experience is a much better teacher than a book (including this one), or a program, but you need to learn the basic information somewhere—which is why a book and/or a program is often helpful when you are learning about your computer.

SAT Instruction. The infamous Scholastic Aptitude Test (SAT) is one of the most important criteria used by colleges considering high school students for admittance. Because of this, the SAT is a matter of great importance to many young people. This is why programs which assist young people with the SAT tend to be very popular. In addition, computers can teach other tests, such as the PSAT, Achievement Tests, or tests at the college level.

Drilling and Practice. The most popular kind of educational program is the "drilling and practice" program, which teaches a person something through repeated instruction and testing. The subject could be mathematics (addition, subtraction, multiplication, geometry), science (formulae, science in history), or any other subject you can imagine.

Educational software is commonly used in the home, but

some schools are starting to purchase these types of programs as supplements to their teaching curricula. At home, another useful type of software available for the 600XL or 800XL is the home finance program. Keeping track of financial records is an excellent application for computers; the 600XL or 800XL is able to help you plan your budgets, balance your checkbook, make investments, and manage your whole "portfolio" and finances for your household. Being a computer, the 600XL or 800XL is a "whiz at math" with a significant amount of memory; the programs using these qualities to keep track of home finances will be very useful indeed.

Even though the 600XL or 800XL can be used with the popular AtariWriter word processor from Atari, Inc., there are additional word processing programs available, which you will see advertised in most general computer magazines. They may have some extra features you might find desirable that AtariWriter doesn't have. Besides the other WP programs, there are "supplemental" programs made for AtariWriter and other word processors which check the spelling of your documents. You can then use your word processor for form letters and other business-related needs without having to worry about misspellings.

Finally, another application made possible through software development for the 600XL and 800XL is the *use of other computer languages.* Currently the 600XL and 800XL not only let a user work with BASIC, but also have the software to support other languages (such as machine language and LOGO). You can learn more about the languages of computers (for there are many of them) and find ways in which programs can operate faster and more efficiently.

The demand for software with the 600XL or 800XL has been very high, but there are always more programs or applications you can create (by making your own programs) that people will find useful. Here is where a great opportunity for you comes in. You can make software for the 600XL or 800XL and sell it to other people. You could be one of the "suppliers of software" for the many who will be wanting it. On the other hand, you might want to program the software yourself and, instead of selling it yourself, find a software publisher to mass produce and sell it for you. This is exactly the subject of the next section, in which we'll explore ways to create programs with your machine—and perhaps later make money with your 600XL or 800XL computer.

GETTING AND DEVELOPING THE IDEA

I have written a large number of nationally marketed programs myself, so people sometimes ask me, "What is the hardest part about making a program?" I reply to their surprise, "Getting the idea." This might not be an absolutely true statement, since debugging a program is also extremely difficult, but I do find that getting and developing an idea for a program is often one of the most difficult parts of programming.

When you are trying to think of an idea for a program, you should think of who is going to be using it. Naturally, if it is for your own use, you should make only what you need. If you are going to be making a program for other people to use, however, you'll probably want to think of an original idea. This can be very difficult indeed, since there are already a great many programs available on the market. The best way to create an original idea for a program, though, is to think of a *need* which someone might want "filled" by a computer. Here are a few types of people, along with a sample program each might find useful:

POTENTIAL USER	USEFUL PROGRAM IDEA
Teachers	Software to keep track of grades
Church Administrators	Program to keep records of donations
Students	Programs to help with SAT test
Small Businessman	Inventory records program
Audiophile	Program to keep track of records

Remember also, especially if you are a young person who enjoys video games, that games are not the only programs you can create for the 600XL or 800XL. Some kinds of programs you could try to include word processors, communications programs, utility (programmer helper) programs, and almost anything else you can devise. Just keep this in mind: *the more original, useful, and needed your program is, the more popular it will be.*

If you are going to create a program simply for your own purposes, then teach yourself more about computers, think of the various features your computer possesses and how to exploit them, and experiment with BASIC keywords and techniques you have *not* used before. You can learn how to program like a professional, since professionals build their

knowledge on their own experience.

The best-selling programs are attractive to large numbers of people; however, you might want to make a program for a more narrow group of people—like your family, your friends, or students who attend a nearby school.

Let's assume that we need to create a program for an introductory computer class. In order to get the idea for this program, we think about what would impress or interest a class about computers. One idea that comes to mind is a short, simple program which lets the computer introduce itself, gets the user's name, and then greets the user by name. This is, of course, a very basic program, but for this example it will suffice.

At this point, we need to develop the program idea. Development involves the *expansion* of your idea, paying attention to minute details, and sometimes drawing a flowchart of your program. Not all of the specifics you write down about your program may make it into the final product. You should have some flexibility in your programming, but you should try to stick to your original ideas as much as possible.

As far as flowcharting is concerned, we aren't going to use a flowchart in our programming example because the program is too short and simple to require one. More complex programs do require flowcharts, but you can probably get away without them until you are making programs larger than, say, forty lines long. Still, flowcharting is good logical exercise and an excellent habit to aquire.

A flowchart is basically a picture of a program's "flow"—the sequence of processes by which it works. A flowchart shows, with words and shapes, the main parts of a program and how they interact with one another. There are five main symbols, shown in Fig. 7-1, which make up most flowcharts.

START/STOP oval. This indicates the beginning and end of a program.

Rectangle. Most of a program's functions are represented within a rectangle, which represents unconditional, "process" action. Such statements as PRINT, READ, clear the screen, and variable assignments can be represented within the rectangle, along with most other statements not including INPUT, END, and IF-THEN.

Diamond. This figure represents conditional action, i.e., decisions made by the computer. IF-THEN statements are contained within diamonds. There are two arrows coming out of the

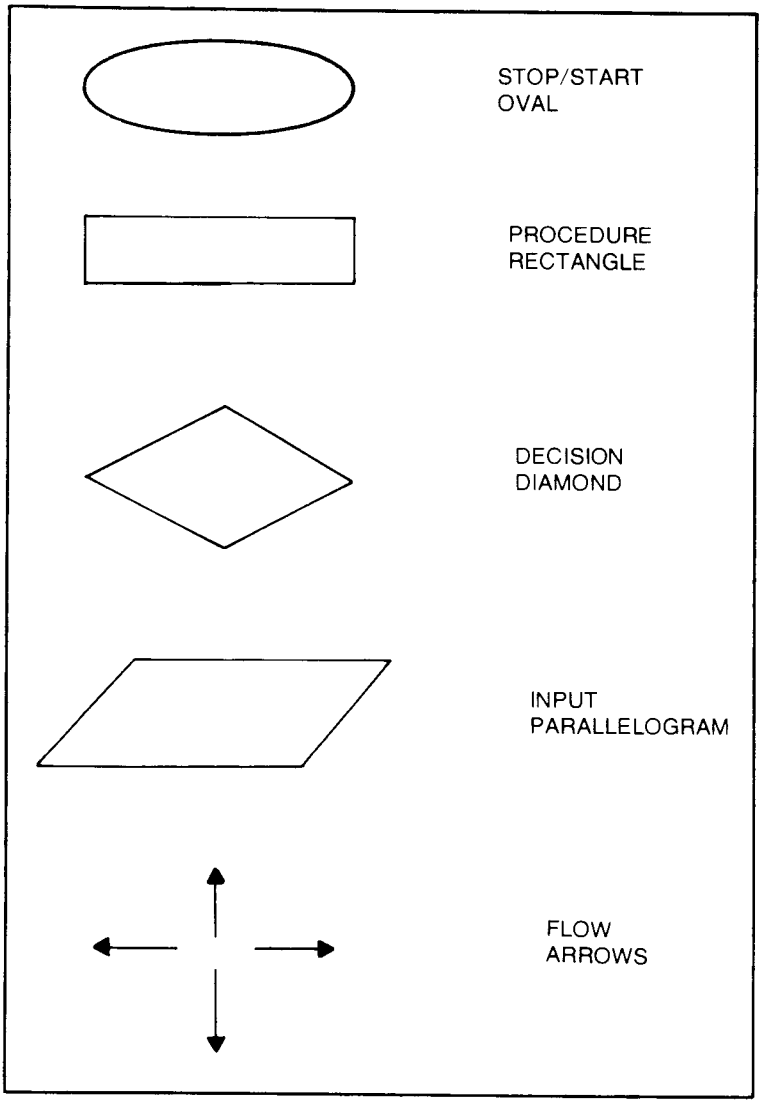


Fig. 7-1. Flowchart symbols.

diamond rather than just one, as with other statements. One arrow represents where program execution will "flow" if a condition turns out to be false, while another arrow is for a true condition.

Parallelogram. For statements involving INPUT, use this figure.

Arrows. The arrows in a flowchart represent the "flow" of a program. Execution can travel only in the direction of the arrow.

Another important part in developing a program is forming the essential details relating to the program. For example, some of the details relating to our simple example in this chapter include:

1) The program will have to introduce itself to the person. This will involve clearing the screen and PRINTing several lines.
2) When the computer asks the user's name, an INPUT statement will be required to get the input.
3) Once the computer knows the person's name, it will have to use the person's name (now represented by a variable) along with some PRINT statements to greet that person.

If you are making a more complex program, such as one involving business or word processing, your "details" can be far more general, such as, "The computer will have to be able to search for words within a document, so I must program it to store a 'search string' and look for it within the document," or, "In the middle of the program, the values will have to be stored on the disk to insure they are not lost." As far as our program is concerned, however, we have created an idea for it and we know the fundamentals parts of the program which must be coded.

PROGRAMMING

The actual programming will be a lot easier if you have a flowchart at hand when you are making a large program. Whatever the size of your project, you should have the details of the program written on paper, as well as a good idea of what keywords and techniques you'll be using in the program. If you are making a large program, try breaking it down into smaller sections, and program these one at a time. In addition, try checking each of these individual sections as you go along to save yourself time in the debugging process. Getting rid of problems now will be much easier than searching through your entire program later for the bugs. Last, be creative, innovative, and as efficient as possible when you are programming, since these three elements are essential to creating a quality program.

Our "simple sample" won't be hard to make. First of all, we need to construct the computer's introduction as follows:

```
10   REM INTRODUCTORY CLASS PROGRAM
20   PRINT"HI THERE.  I AM AN ATARI COMPUTER.  I AM NEW"
30   PRINT"AROUND HERE, SO I WAS HOPING YOU COULD TELL"
```

The first line is a REMark to show the purpose of the program. In lines 20 and 30, the computer introduces itself. At this point, we can make the part which allows INPUT from the user so that he can identify himself.

```
40   INPUT"WHAT IZ YOUR NAME";N$
```

Finally, the computer greets the person by name:

```
50   PRINT"HELLO THERE, ";PRINT N$
```

Don't be alarmed if you see a few errors in the program above. This is intentional since we need to see how we go about debugging a program in the next section.

DEBUGGING

The most grueling and boring step of making a program, as well as the most exciting and rewarding, is the debugging process. It is boring because you have to search through your program to find any problems or potential errors, and it is exciting because you will feel a surge of victory every time you conquer a bug. Looking for those elusive bugs involves testing every possible input the user could give the computer, making the program do everything it is capable of doing, and trying your best to make the program fail. You will probably be somewhat cautious while testing your program, since nobody likes to find more problems to solve. Do not simply give up, however, assuming that you have conquered all the bugs. You must give your program a real workout to assure top performance. Spend all the time you can with it, since there may still be one problem which a user could discover later. This could be rather embarassing.

Our program has a few problems in it. First of all, it should do the following:

```
HI THERE.  I AM AN ATARI COMPUTER.  I AM NEW
```

```
AROUND HERE, SO I WAS HOPING YOU COULD TELL
ME: WHAT IS YOUR NAME? TIM KNIGHT
HELLO THERE, TIM KNIGHT
```

However, if you run the program, you will find it does this:

```
HI THERE.  I AM AN ATARI COMPUTER.  I AM NEW
AROUND HERE, SO I WAS HOPING YOU COULD TELL
WHAT IZ YOUR NAME? TIM KNIGHT
HELLO THERE,
TIM KNIGHT
```

As you can tell, we have a few mistakes which need correcting. To begin with, the third line of text is not right. Change the WHAT IZ YOUR NAME to ME: WHAT IS YOUR NAME. Also, the name is printed on the second line rather than on the same line as HELLO THERE. To correct this, you can either put a semicolon after the end quote following HELLO THERE, or you can simply eliminate the second PRINT statement in line 50. The line would then read PRINT"HELLO THERE, "N$.

This program wasn't very difficult to debug since it was so small, but most programs are rather tedious and mind-boggling when you are debugging them. Remember these important things when you are debugging any program:

1) For a program that requires input, try inputting anything you can imagine. If the program wants numbers, give it letters, names, symbols, and anything else you can type. If a computer wants you to use the "6" key to move the cursor right, keep pressing it to see if an error will occur when the cursor tries to move off the boundaries of the screen.
2) Try every incorrect method of using a program to make sure that it is foolproof.
3) Get other people to use your program. They are much better judges of it than you are. They can also provide helpful constructive criticism and comments on the program, and can alert you to problems existing in the software.

Once you have finished programming and debugging a piece of software, you should write some documentation (instructions) for it. Simply type out the basic facts a person needs to know to use your program, and mention some ways in which the program can be used. If you are doing the program for a software company, you should make the documentation itself an excellent work. Since you probably won't be doing that for a while, there's no need yet to concentrate on writing anything but the essentials.

MAKING PROGRAMS AND MONEY WITH YOUR ATARI

Many people (including a large number of young people) have made something of a business out of writing software for computers. In fact, being a professional programmer can be a very profitable job, even if you just work out of your home producing one or two programs a year. Until now the software market has been pretty well saturated for most computers, but newly introduced machines offer an excellent opportunity for programmers to write more software for a variety of applications. This opportunity exists with the 600XL or 800XL, and you could make yourself quite a lot of money with your computer through software development.

There are a number of ways to make money with computers (see Fig. 7-2). Writing software reviews for computer magazines, writing articles, creating computer-related books, teaching computer classes, or serving as a computer consultant to a person or a business are just a few. However, the particular way of making money with your 600XL or 800XL I am discussing now—making software—probably offers more rewards and excitement than any other computer-related business opportunity.

The profits of making software come in a variety of forms. First of all, writing a program will inevitably put you in the position of being an "exterminator," since you'll have to seek out and destroy all the bugs in the program you create. The profit from this, though, is the good feeling you have when you finally remove a bug from your program. Until you actually find and fix a difficult mistake in one of your own creations, you won't be able to understand the feeling of accomplishment your achievement can give you. After you have, you'll want to keep searching and stamping out those bugs, no matter how frustrating and time-consuming they might be.

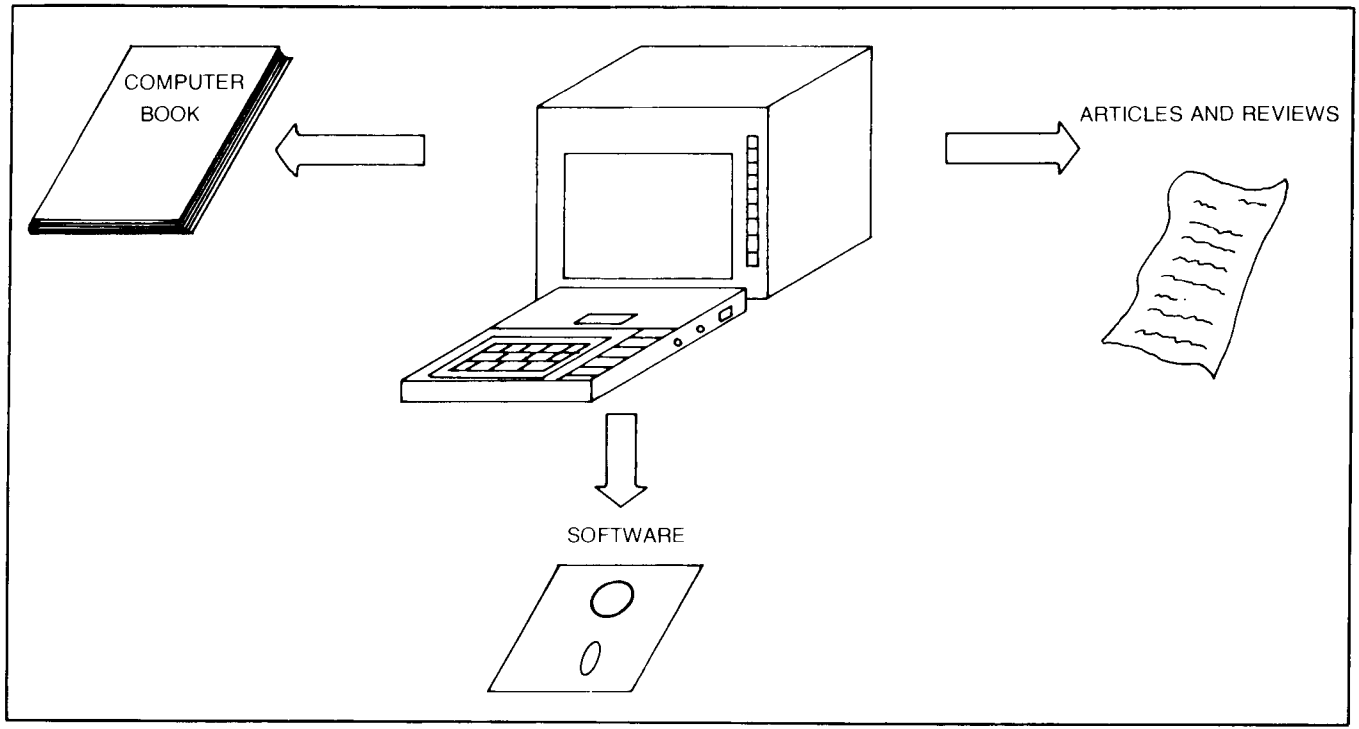


Fig. 7-2. Ways to make money.

Another profit which comes from making programs, naturally, is the money you receive. If you give your program to a software publisher, which is probably the best decision, you will receive a certain royalty for every copy of your program that is sold. For instance, assume you have made a great home finance manager program that is being sold for $30 per copy. Assume again that you receive 25 percent of the *wholesale* price of your program (about $15), making your royalty $3.75. If 10,000 600XL or 800XL users purchase your program, this would result in a profit of $37,500 for you—which is an excellent compensation for your work as a programmer. Of course, you shouldn't always expect royalties this high, but some people have made millions of dollars on their programs. (One person I know made a million with one arcade game.) Many other programmers have six-figure incomes for their work. The more popular the 600XL or 800XL becomes, and the better your program is (in addition to how good your software publisher is), the more money you are likely to receive for your program.

The method by which you create a program was discussed earlier in this book, but here are a few more pieces of advice when creating your program for the software market:

1) Make the documentation as complete and easy to understand as possible. This will assist the novice 600XL or 800XL user in utilizing your program.
2) Test the program as thoroughly as you can before sending it to a software publisher. Have other people use your program, and get their constructive criticism of how your program can be improved, what problems exist in the program, and what can be done with the documentation to make the program easier to use as soon as a person purchases it.
3) When thinking up your original idea for the program, make it as original as possible and as attractive to a buyer as you can. Imagine yourself in the position of someone who is looking at a software package and has not tried the program, but who wants to find something that is entertaining, educational, or very useful.

The important step now is to get in contact with the software publishers who interest you, see if they like your *idea* for a

program, and then get to work on the actual software. Develop it and have the publisher market it for you.

THE FUTURE OF COMPUTERS

Since computers have changed so much over the past few years, it's difficult to say what will be happening to the ATARI 600XL, ATARI 800XL, other home computers, and the field of high technology in general. However, I do have some suggestions as to what will probably be occurring over the next couple of years in these areas. These predictions may prove useful to you in some way.

First of all, computers aren't likely to drop in price much more than they already have. When personal computers were introduced, they cost $2,000 or more, but a few years later the price fell to $500, $300, and then to $100 or less. Unfortunately, since the price fell so much, the cheapest personal computers became poorly constructed toys. The price increased, along with the quality, back to $200. At this price, personal computers were affordable and still of a quality high enough to satisfy the consumer. Prices stayed around $200 to $700 for home computers, but the features began to increase. The result, up to now, has been the 600XL or 800XL computer—low in price, but with a lot of fantastic features which would have cost you thousands of dollars a few years ago. This trend is likely to continue. You'll probably see more inexpensive and feature-packed computers appearing, in addition to low-cost extra peripherals and software being made available for the 600XL or 800XL.

Another important trend is that software is getting easier to use and more practical. This will result in computers being made even more useful than they already are, with programs being written in such easy to understand terms that even the novice will be able to utilize all kinds of software. A good example is Atari-Writer, which is a program made simple to use by extensive prompting from the computer.

Finally, computers will be expanding into what I like to call *probots,* which is short for personal robots (Fig. 7-3). Personal robots are an extension of the home computer, with all the powers of a computer like the 600XL or 800XL, but with the additional utility of motion and the power to manipulate objects. Soon your 600XL or 800XL computer may be able to control a personal robot and make it guard your house, watch for and extinguish fires, and entertain guests. The personal robotics

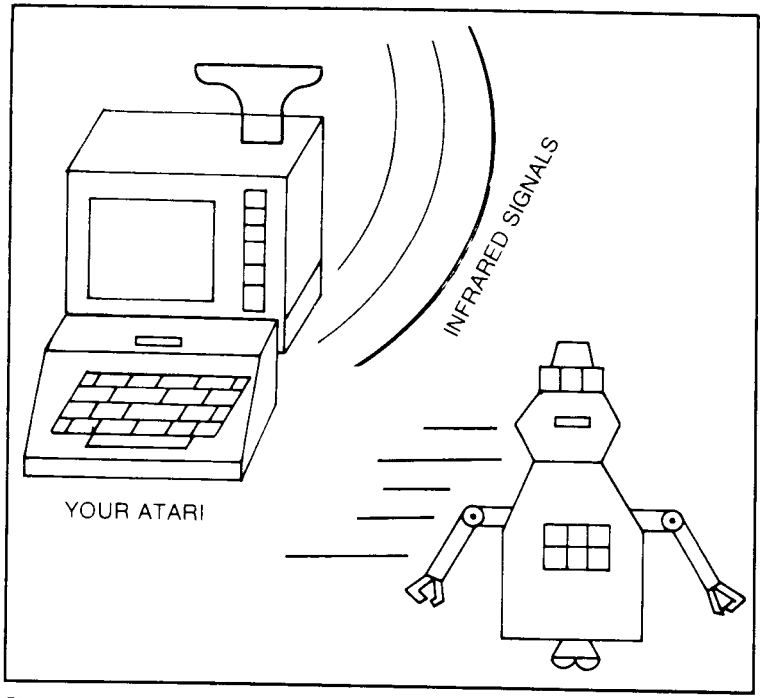


Fig. 7-3. A probot of the future.

revolution is likely to be as influential as the personal computer revolution, so it will be exciting to see what developments occur in that new field.

Your 600XL or 800XL computer is a powerful tool and a wonderful extension of your own mind. Use it like a tool, and let it entertain you, make you more efficient, and keep your records in order. The personal computer revolution has brought a lot of changes, including this inexpensive and powerful system called the 600XL or 800XL. Learn to program it well through your own experiences, and use it as often as you can for as many varied applications as you can find—to discover how useful and enjoyable the ATARI 600XL or 800XL computer can truly be.

Glossary

Array—A set of variables set apart from one another by numerical subscripts. Examples: A(1), B$(20), C(50). Before you use an array, you must set it up with the DIM statement.

ASCII—A standardized coding system which assigns corresponding code numbers to letters, numbers, and symbols.

Binary—The number system computers use, composed of zeros and ones.

Bit—The smallest unit of information the ATARI can hold. Bit is short for *binary digit,* since a bit can be either one or zero.

Bug—A problem in a program.

Byte—The basic unit of information in the computer. A byte is a unit of memory equal to eight bits.

Command—A word you give to the ATARI computer so that it will carry out some function.

Disk—A piece of Mylar plastic on which you can permanently store programs, data, and other kinds of information.

Disk drive—The device you use to store and retrieve programs and information. The disk drive is faster and more efficient than the tape recorder.

Display—The television set, monitor, screen, video display, or whatever you want to call the device that shows you the information of the computer on a cathode-ray tube.

Edit—To modify or change information.

Error—A mistake in a program, or in one of your commands to the computer.

Execute—To begin running (using) a program.

Graphics—Visual information in the form of pictures such as bar graphs, spaceships in a game, or a piece of "computer art" formed by the ATARI's advanced high-resolution graphics. The ATARI can also support low-resolution graphics, which aren't as well-defined but can be displayed in a wider variety of colors.

Hardware—The electronic components of a computer system, such as the printer, the keyboard, the microprocessor, and so on. Hardware can be physically damaged if mistreated, so treat your ATARI with care.

Integer—A whole number without a decimal part. 5, 2, 6, and 2,034 are integers, while 2.53245 is not.

Joystick—The device usually used in games to move your "man" around the screen. Joysticks are attached to the keyboard and can be used to indicate direction. Because of this, they are commonly used in arcade-type games.

K—The name given to designate a certain amount of memory. One K is equal to 1,024 bytes. The ATARI 600XL has 16K, which equals 16,384 bytes of memory.

Keyboard—The part of the ATARI Computer System on which you can type information for the computer to use.

Machine Language—An extremely fast means by which the computer can carry out programs, since machine language is the actual "language of the computer"—as opposed to BASIC, which must be interpreted by the computer and translated by the ATARI into its own machine language.

Memory—The computer's information storage center.

Peripheral—An extra device for your computer, such as a printer, a modem, a disk drive, or a joystick.

Printer—The device which gives you a "hardcopy" (printed version) of information, such as a letter, a listing of a program, or an essay.

Reserved word—A word which can be used only by the ATARI. You cannot use it for variable names or uses other than giving the ATARI commands. For example, you couldn't let a variable named LIST equal 10 since the name LIST is used by the ATARI to list out programs. It is a *reserved word.*

Software—The programs the ATARI uses. When you turn the computer off, any software (programs) in the computer will be erased. In order to save programs for later use, store them onto a disk or a cassette tape.

String—A series of characters and/or symbols assigned to a variable called a *string.* String variables are represented by a dollar sign ($) following them. A$, Y1$, and UU$ are all string variables, which can equal such things as "123," "YOU THERE," and "ATARI."

Text—Words and letters you can understand, such as this sentence.

Index

BASIC statements and functions are indexed by first usage in a program.